НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
ІМЕНІ ІГОРЯ СІКОРСЬКОГО»

Факультет інформатики та обчислювальної техніки

Кафедра технічної кібернетики

«На правах рукопису»
УДК 004.043

«До захисту допущено»

Завідувач кафедри
_____ І.Р. Пархомей
(підпис)
“___”_____2019 р.

Магістерська дисертація

на здобуття ступеня магістра

зі спеціальності 126 «Інформаційні системи та технології»

на тему: Дослідження питань підвищення швидкодії роботи веб додатку, на прикладі системи реєстрації абітурієнтів

Виконав: студент другого курсу, групи ІК-71мн
(шифр групи)

Белоус Роман Володимирович _____
(прізвище, ім’я, по батькові) (підпис)

Науковий керівник доцент, к.т.н., доцент Крилов Є.В. _____
(посада, науковий ступінь, вчене звання, прізвище та ініціали) (підпис)

Консультант _____ Лісовиченко О.І. _____
(назва розділу) (науковий ступінь, вчене звання, , прізвище, ініціали) (підпис)

Рецензент _____ _____
(посада, науковий ступінь, вчене звання, науковий ступінь, прізвище та ініціали) (підпис)

Засвідчую, що у цій магістерській дисертації немає запозичень з праць інших авторів без відповідних посилань.
Студент _____
(підпис)

Київ – 2019 року

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
ІМЕНІ ІГОРЯ СІКОРСЬКОГО»

Факультет інформатики та обчислювальної техніки

Кафедра технічної кібернетики

Рівень вищої освіти – другий (магістерський)

Спеціальність 126 «Інформаційні системи та технології»

ЗАТВЕРДЖУЮ
Завідувач кафедри
_____ І.Р. Пархомей
(підпис)
«___»_____2019 р.

ЗАВДАННЯ
на магістерську дисертацію студенту
Белоусу Роману Володимировичу
(прізвище, ім'я, по батькові)

1. Тема дисертації «Дослідження питань підвищення швидкодії роботи веб додатку, на прикладі системи реєстрації абітурієнтів»,

науковий керівник дисертації _____доцент, к.т.н., доцент Крилов Є. В.____,
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

затверджені наказом по університету від «13» березня 2019 р. №877-С

2. Термін подання студентом дисертації _____20.05.2019_____

3. Об'єкт дослідження *процес роботи системи управління навчальним закладом.*

4. Предмет дослідження *технології оптимізації швидкості роботи системи.*

5. Перелік завдань, які потрібно розробити *огляд подібних веб додатків їх переваги та недоліки; вибір технологій для розробки системи та їх опис; розробка веб додатку для навчального закладу та керування процесом;*

тестування та збір даних по роботі веб додатку; оптимізація швидкості роботи системи навчального закладу.

6. Орієнтовний перелік ілюстративного матеріалу – шість плакатів

7. Орієнтовний перелік публікацій - дві публікації

8. Консультанти розділів дисертації

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
Перевірка на співпадіння	Лісовиченко О.І		

9. Дата видачі завдання _____

Календарний план

№ з/п	Назва етапів виконання магістерської дисертації	Термін виконання етапів магістерської дисертації	Примітка
1	Аналіз предметної області та постановка задачі	05.02.2019 р.	
2	Вибір технологій для розробки веб додатку	11.02.2019 р.	
3	Проектування бази даних	18.02.2019 р.	
4	Розробка основних модулів веб додатку	22.02.2019 р.	
5	Оптимізація роботи веб додатку	01.03.2019 р.	
6	Тестування розробленої веб додатку	11.03.2019 р.	
7	Оформлення пояснювальної записки до дипломної роботи	18.03.2019 р.	
8	Висновки	5.04.2019 р.	

Студент _____ Белоус Р.В.
(підпис) (ініціали, прізвище)

Науковий керівник дисертації _____ Крилов Є. В.
(підпис) (ініціали, прізвище)

АНОТАЦІЯ

У роботі було розглянуто створення та оптимізацію швидкодії роботи системи дистанційного навчання. Провівши аналіз різних веб додатків навчальних закладів та визначавши недоліки, переваг було прийнято створити нову систему управління навчальним процесом, врахувавши всі аспекти. Як результат виконання дипломної дисертації – створено та оптимізовано швидкість роботи веб додатку, за допомогою зменшення навантаження на сервер, інтегрування нового функціонал кешування, оптимізації структури бази даних, медіа файлів та зображень. Було прийнято рішення перенести на окремий, надійний VPS сервер з технологією масштабування, для уникнення проблем з потужністю, так як веб додаток постійно переобуває в процесі збільшення і стає необхідно більше ресурсі на його роботу.

Ключові слова: оптимізація швидкості роботи веб додатку, кешування, React , PHP, MVC, MySQL, VPS сервер.

Розмір пояснювальної записки – 89 аркуші, містить 30 ілюстрацій, 24 таблиці, 4 додатків.

ABSTRACT

The research have evaluated creation and optimization of distance education information system responsiveness. Conducted analysis of different web applications provided by educational institutions has identified flaws that justified redesign and creation of new information system that coordinates education management with all details taken into account. As a result of this research, web application has been created with optimized responsiveness as a result of following enhancements: decreased server load, integration of new caching functionality, optimization of database relational schema, media files and images. The decision has been made to move the infrastructure into a reliable VPS server with auto-scaling capabilities to avoid issues with capacity, since this web application is constantly facing increased loads that demand increased computation power to process requests.

Keywords: web application optimization, caching, React, PHP, MVC (Model-View-Controller), MySQL, VPS.

Explanatory note size: 89 pages containing 30 illustrations, 24 tables and 4 supplementary materials.

ПОЯСНЮВАЛЬНА ЗАПИСКА

до магістерської дисертації

на тему: Дослідження питань швидкості роботи веб додатку, на прикладі системи навчального закладу

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ 9
ВСТУП 10
1. АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ ТА ПОСТАНОВКА ЗАДАЧІ 12
1.1. Об’єкт та предмет дослідження 12
1.2. Аналіз існуючих рішень 13
1.3. Постановка задачі 19
Висновки до розділу 19
2. РОЗРОБКА СИСТЕМИ УПРАВЛІННЯ НАВЧАЛЬНОГО ЗАКЛАДУ 21
2.1 Концепція побудови структури системи 21
2.2 Технології та інструменти розробки 24
2.2.1 Мова розмітки HTML та фреймворк Bootstrap 4 24
2.2.2 JavaScript та фреймворк ReactJS 28
2.2.3 PHP та фреймворк Laravel 29
2.2.4 MySQL 31
2.3 Проектування веб додатку 31
2.3.1 Проектування модулів веб додатку 32
2.3.2 Проектування ролей веб додатку 35
2.4 Структура бази даних 39
2.4.1 Опис та зв’язки таблиць 39
2.4.2 Створення міграцій 41
3. ОПТИМІЗАЦІЯ ШВИДКОСТІ РОБОТИ ВЕБ ДОДАТКУ 51
3.1. Оптимізація бази даних 51
3.1.1 Оптимізація потоків SQL-запитів 51
3.1.2. Критерії оптимізації 53
3.2. Виділення оптимізуючих класів задач і розробка методу 57
3.2.1. Запити з зовнішніми ключами 58
3.3 Метод кешування індексів 61
3.4 Оптимізація клієнтської частини 63
Висновки до розділу 65

4. МАРКЕТИНГОВИЙ АНАЛІЗ СТАРТАП-ПРОЕКТУ 67
4.1 Опис ідеї проекту 67
4.2 Технологічний аудит ідеї проекту 69
4.3 Аналіз ринкових можливостей запуску стартап–проекту 70
4.4 Розроблення ринкової стратегії проекту 77
4.5 Розроблення маркетингової програми стартап–проекту 80
Висновки по розділу 82
ВИСНОВКИ 83
ПЕРЕЛІК ПОСИЛАНЬ 84
ДОДАТКИ 85
ДОДАТОК А 86
ДОДАТОК Б 87
ДОДАТОК В 88
ДОДАТОК Г 89

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ

СУБД – система управління базою даних

PHP – hypertext preprocessor

MVC – model-view-controller

СУБД – система управління базою даних

SQL – мова програмування для баз даних

RJS – JavaScript бібліотека

ОП – оперативна пам’ять

JSON – JavaScript object notation

ВСТУП

Поява потужних комп'ютерних мультимедійних систем, нових технологій, інтерактивного процесу, сучасного програмного забезпечення давно стало основою прогресивного розвитку дистанційного навчання в різних навчальних закладах, як в Україні так і в світі.

Дистанційний вид навчання на сьогоднішній день, дає нам можливості створення нової систем масштабного самонавчання, загального обміну інформацією, незалежно від місця чи часу знаходження людини. Тим більш, системи дистанційної освіти дають різні рівні можливостей людям незалежно від стану в соціальному середовищі, як усередині України, так і а інших країнах реалізувати одне за головних прав людини – права на освіту і отримання нової інформації.

Дистанційні освітні технології все ширше входять в систему освіти. Змінюється характер міжособистісних комунікацій в сучасному світі, з'являється об'єктивна необхідність активізації використання дистанційних освітніх технологій в навчально-виховному процесі, серед яких можна виділити презентації, соціальні мережі, веб-квест, кейс-технологію, електронне середовище навчання і т.п. Всі вони значно змінюють функції і статус викладача в навчальному процесі, який з джерела знань стає посередником, модератором, комунікатором і навіть техніком-організатором.

Викладач звільняється від монотонної роботи при організації навчального процесу, це дає можливість створити багатий довідковий та ілюстративний матеріал, представлений в самому різноманітному вигляді: текст, графіка, анімація, звукові і відео елементи. Інтерактивні комп'ютерні програми активізують всі види діяльності людини: розумову, мовну, фізичну, що прискорює процес засвоєння матеріалу.

Навчальні заклади повинні розширювати можливості професійного розвитку населення, коли творчий потенціал і людські якості членів суспільства стають головним ресурсом ефективного розвитку, успіху в конкуренції на світових ринках, безпеки країни і високої якості життя громадян.

Актуальність дистанційного навчання полягає в тому, що в умовах техногенної цивілізації традиційні моделі організації навчального процесу не в змозі задовольнити потреби в освіті значної частини населення. У цих умовах система дистанційного навчання, що забезпечує використання новітніх технологічних засобів для доставки інформації та навчальних матеріалів безпосередньо споживачеві незалежно від його місцеперебування, стає невід'ємною, конкурентної частиною освітнього простору.

Система дистанційного навчання повинна бути гнучкою і в міру складною. Занадто складна система дистанційного навчання шкодить сама собі, так як над діями самої системи може бути втрачений контроль.

В системі викладені навчально-методичні комплекси за програмами курсів навчання, які містять електронні підручники, курси лекцій, завдання, тести для самоконтролю та контрольні тести за підсумками навчання. Слухачам дистанційних курсів надається можливість отримання консультацій у викладачів очно і з мережевої взаємодії. Саме дистанційне навчання дає змогу найбільш універсально, гнучко і адекватно реагувати на потреби суспільства і забезпечити реалізацію конституційного права на освіту кожного громадянина країни. Для коректної роботи дистанційної системи навчального закладу необхідно проводити її систематичний аналіз, а саме:

- оптимізацію бази даних системи;
- навантаження роботи серверу, збільшення ресурсів при необхідності;
- періодичне оновлення програмного забезпечення;
- моніторинг та тестування швидкості роботи серверу системи;

Аналіз результатів дає можливість оцінити ефективність роботи системи дистанційного навчання та визначити в ній наявні резерви ефективності так як дистанційне навчання є та буде залишатись в 21 столітті, як найефективніша сучасна система підготовки і безперервної підтримки високого кваліфікаційного рівня фахівців, що збільшує конкурентоспроможність на ринку праці на міжнародному рівні.

1. АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ ТА ПОСТАНОВКА ЗАДАЧІ

1.1. Об’єкт та предмет дослідження

Дослідженням даної магістерської роботи направлене на оптимізацію швидкості роботи веб додатку навчального закладу. На сьогоднішній день по всьому світу ми спостерігаймо тенденцію дистанційного навчання. Кожен вуз, університет, школа, ліцей та гімназія інтегрують в процес навчання сучасні технології дослідницької діяльності, проектної, інформаційно-комунікаційні, особистісно-орієнтовані та інші. Перераховані вище технології спрямовані на розвиток в учнів різних особистих якостей. Одночасно з цим розвивають вміння нестандартно вирішувати певні завдання, відійти від шаблонної діяльності, бути більш залученими в процес навчання, ніж при стандартній технології навчання, підвищувати свої комунікативні навички, що має вести до гармонійного розвитку особистості, придбання необхідних знань. У цій ситуації прийнято завдання створити нову систему управління навчального закладу, розробивши сучасну оптимізовану структуру бази даних для великої кількості користувачів. Система має мати чітко продуманий інтерес, орієнтований на різні вікові групи, від дітей та до викладачів. Почавши розробку веб-додатку потрібно провести аналіз роботи подібних, існуючих рішень. Визначити їх преваги та недоліки роботи. Зрозуміти, що саме сповільнює їх роботу, чому так відбувається, які технології краще використовувати, який функціонал необхідно інтегрувати, та який не потрібно. Визначити основні аспекти , за допомогою отриманих даних зробити висновки та реалізувати новий веб додаток.

Об’єкт дослідження – процес роботи веб-додатку для навчання студентів та керування навчальним процесом та час виконання запитів користувачів.

Предмет дослідження – методи та технології оптимізації роботи веб додатку, які б забезпечили його належну швидкодію.

1.2. Аналіз існуючих рішень

Сьогодні як в Україні так і всьому світі стає популярним онлайн навчання, результатом чого є створення різних систем та веб додатків для отримання нових знань. Ці системи частіш за все створюються для приватного навчання, але держані заклади також розвиваються у даному напрямку. Розібравши детально дану тенденцію можна сказати, що кожен заклад пропонує свій стиль навчання, різні методичні та відео матеріали та іншу інформацію направлену на збільшення знань свої студентів. Одним с таких додатків можна знайти у приватної компанії SwiftBook зображена на рис. 1.1, яка спрямована на доволі молоду мову програмування від американського бренду Apple – Swift, що витіснила мову Objective-C.

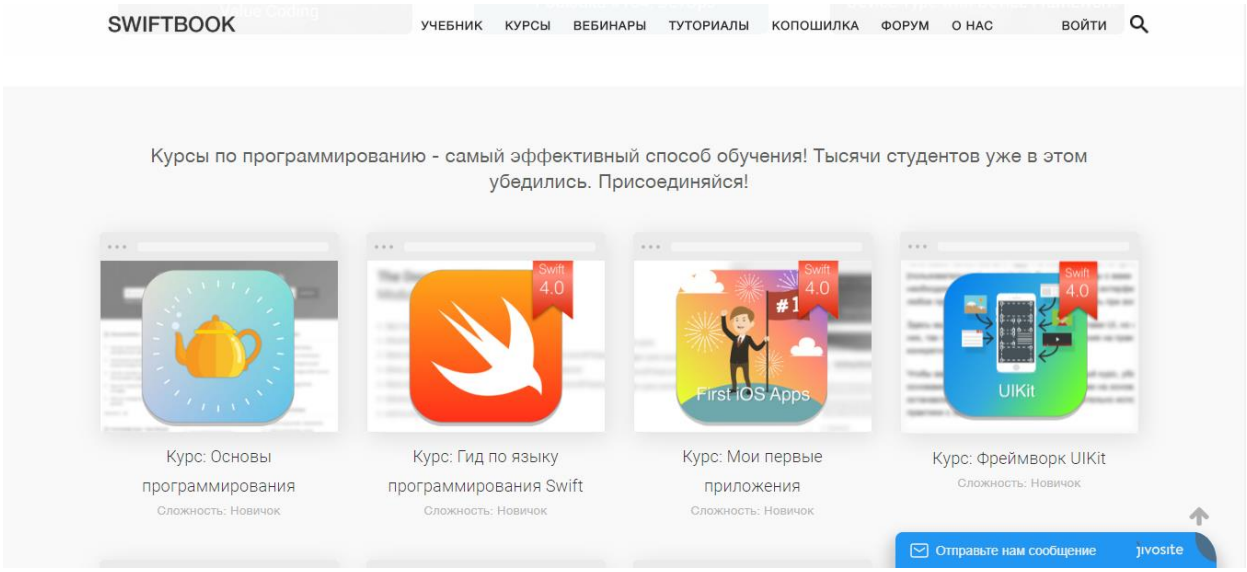

Рисунок 1.1. Веб додаток навчального закладу SwiftBook

Розглянувши веб додаток можемо сказати, що він має мінімалістичний дизайн. Перше, що можна побачити - це меню вгорі сторінки, далі верхньому лівому кутку бачимо кнопку “Увійти” за допомогою якої можна потрапити у особистий кабінет або створити його. По центу відображені великі динамічні елементи - блоки з іконками та назвами курсу, що можна прослухати . При виборі певного курсу користувач буду пере направлений на нову сторінку, де буде детально описано зміст курс, загальні інформація, тип навчання, його тривалість, особливості та інші деталі, для старту освоєння нових знань користувача рис. 1.2.

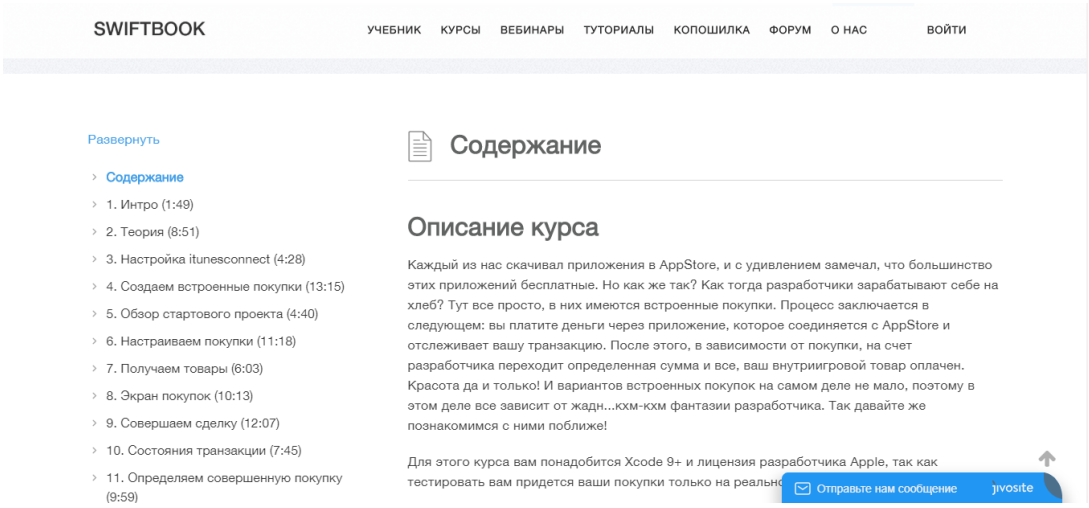

Рисунок 1.2. Представлення курсу в системі SwiftBook

Після реєстрації вданій системі, користувачу приходить на електронну пошту лист для підтвердження та завершення процесу створення нового особистого кабінету. Далі після активації користувачу пропонують вибрати підписку на один місяць та почати навчання в даному веб додатку. Гарним модулем даного веб додатку є модуль за допомогою якого користувач може поставити питання вчителю, та отримати відповідь рис. 1.3. Перевагою є те що відповіді може бачити будь який користувач, що економить час пошуку на вирішення питання.

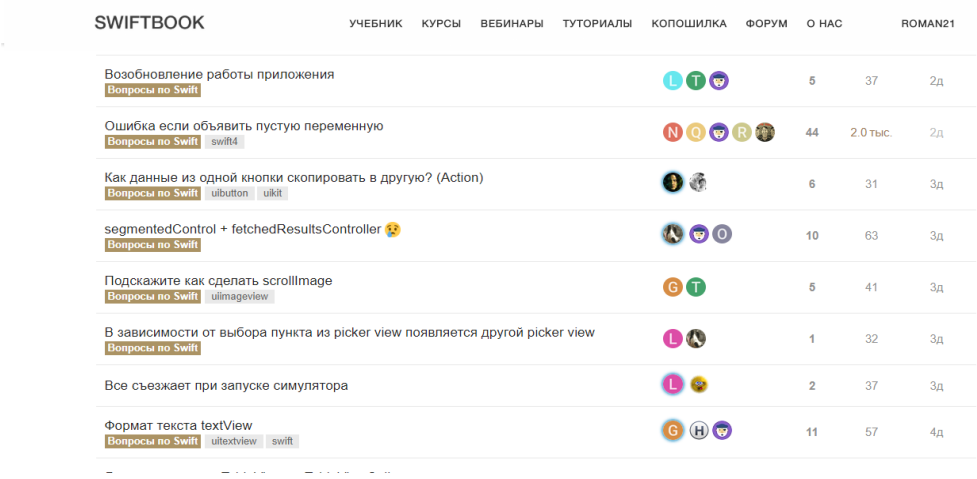

Рисунок 1.3. Модуль поставлення питань у веб додатку SwiftBook

До недоліки даної системи можна віднести мінімалістичний дизайн, тому що подібна система повинна мати більш наповнений інформацію інтерфейс, для більш зручного користування. Також дана система має проблеми з мобільною версією, та погано відображаються в більш старих браузерах. Якщо звернути

увагу на швидкість роботи сервера та провести невеликий аналіз та можна сказати, що даний веб додаток потрібно оптимізувати, для коректної роботи та зручного користування (табл. 1.1).

Таблиця 1.1. Результати роботи веб додатку SwiftBook

Тип проблеми	Для комп'ютерів	Для мобільних
Час завантаження першого контенту	2,7 сек.	8,3 сек
Індекс швидкості завантаження	5,9 сек	20,9 сек
Час завантаження достатньої частини контенту	3,8 сек	12,8 сек
Час закінчення роботи ЦП	4,5 сек	17,2 сек
Час виконання коду JavaScript	4,1 сек	15,7 сек
Робота системи в основному потоці	6,7 сек	23,1 сек
Розмір структури DOM	2 302 вузла	2 281 вузла
Надмірне навантаження на мережу	8 692 КБ	8 678
Помилки в консолі	10	10

З даних наведених в таблиці можна сказати наступне - веб додаток орієнований лише на людей, що користуються лише комп'ютером, тому як дані для мобільної версії кардинально відрізняються від даних десктопної. Основу увагу потрібно звернути на швидкість завантаження, особливо для мобільної версії та кількість JavaScript помилок. В ідеалі помилок не повинно бути взагалі так як це погіршує швидкість роботи веб додатку та взаємодії с користувачем. Так можна сказати що на даній сторінці, яку було проаналізовано, відображені лише зображення курсів та їх назви, а у випадку додавання інших елементів які важили б набагато більше такі наприклад як відео, презентації та методичні матеріали, то сторінка завантажувалась довше іншого і виникли б проблеми, з ефективністю даного веб додатку та користуванням. Тому дану систему рекомендовано оптимізувати по всім пунктам що були наведені в табл. 1.1.

Також можна розглянути наступну систему управління дистанційного навчання – Prometheus. Даний веб додаток був створений у 2014 році за допомогою трьох молодих та сучасних викладачів провідних вузів України. Потрапивши на головну сторінку платформи Prometheus рис. 1.4, користувач бачить перед собою меню, де можна отримати повну інформацію про курси, почати стежити за веб додатком в різних менеджерах та увійти то особистого кабінету. По центру сторінки можемо знайти поле за допомогою якого користувач має можливість відшукати та підібрати для себе відповідний курс – це доволі зручний та сучасний функціонал. Дизайн виконаний у сучасному стилі з використанням популярних технологій та фреймворків.

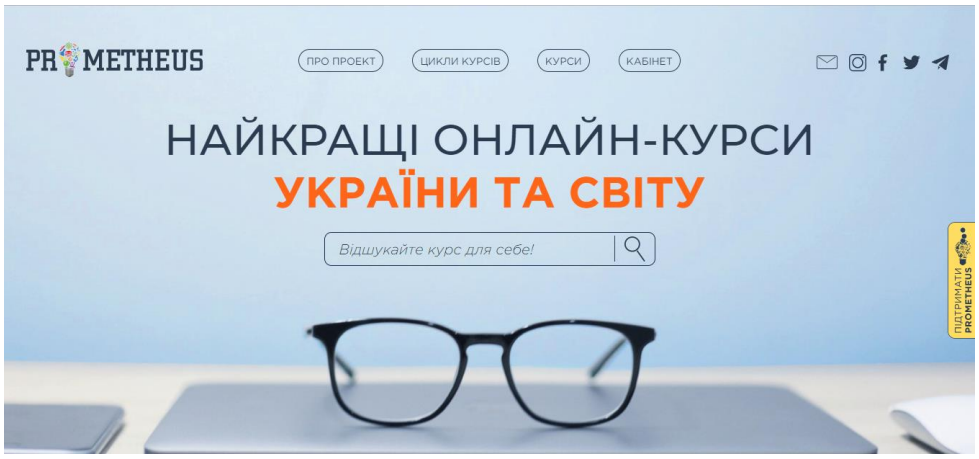

Рисунок 1.4. Головна сторінка веб додатку Prometheus

Перейшовши на вкладку кабінет, користувачу необхідно чи авторизуватись чи пройти реєстрацію. Після потрапляння до особистого кабінету користувач бачить особистий профіль, та може його редагувати, що як на мене не правильно, так як було доцільніше після авторизації в системі перенаправляти користувачів або на головну сторінку, або на сторінку з можливим пошуком курсів, що дало б зекономити час та зацікавленість користувачів. Тому цей етап можна віднести до недоліків. Далі користувач може перейти до пошуку необхідного курсу, отримати всю необхідну інформацію (методичні матеріали, відео, лекції, презентації) та почати його вивчати рис. 1.5.

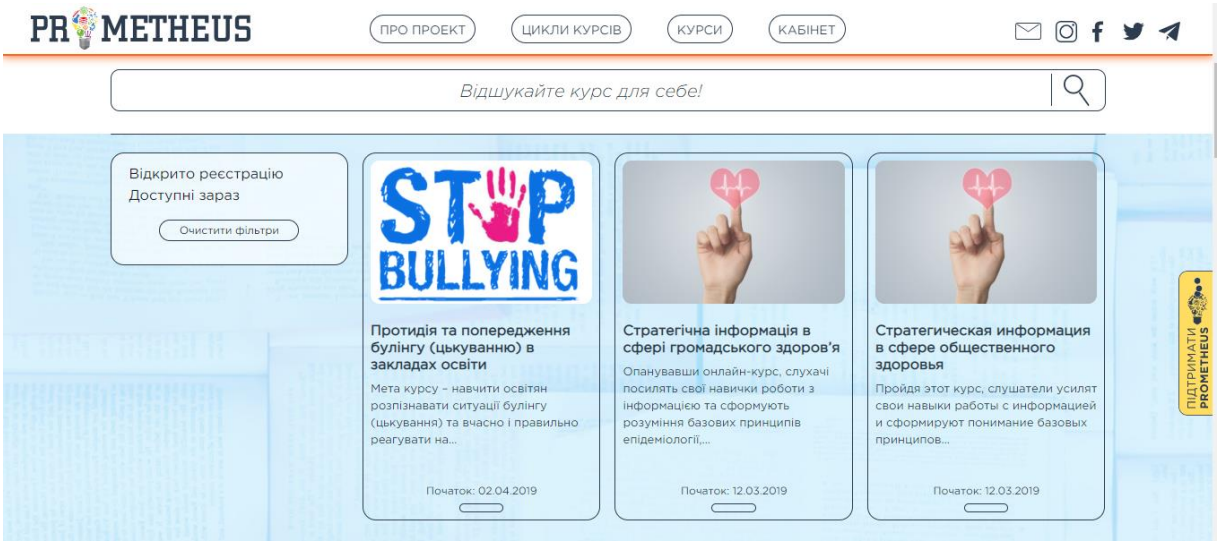

Рисунок 1.5. Сторінка пошуку курсів веб додатку Prometheus

Після потрапляння на необхідний курс користувач переходить на сторінку рис. 1.6, де більш детально розповідається про що даний курс та може знайти кнопку за допомогою якої зареєструються на даний курс. Далі користувач бачить всю інформацію про курс, а саме:

- Назву та інформацію про курс;
- Початок проведення курсу;
- Команду курсу;
- Зміст курсу;
- Обговорення курсу;
- Список термінів;
- Інформацію про етапи проходження курсу;

Розпочинається будь який курс з інструкції про користування даним курсом, де детально розписані бали, які студент може отримати на протязі проходження всього курсу за певний інтервал часу, технічна документація курсу, рекомендації щодо перегляду відео уроків, терміни та рекомендації іноземною мовою. Перейшовши до першого заняття користувач отримує методичні матеріали та змогу переглядати відео до першого уроку, та почати своє навчання за допомогою даного веб додатку Prometheus, не закінчивши попереднє заняття користувачі не

можуть перейти до наступного уроку, що є великою перевагою у навчанні, так як студент що хочу отримати сертифікат, повинен пройти весь курс навчання.

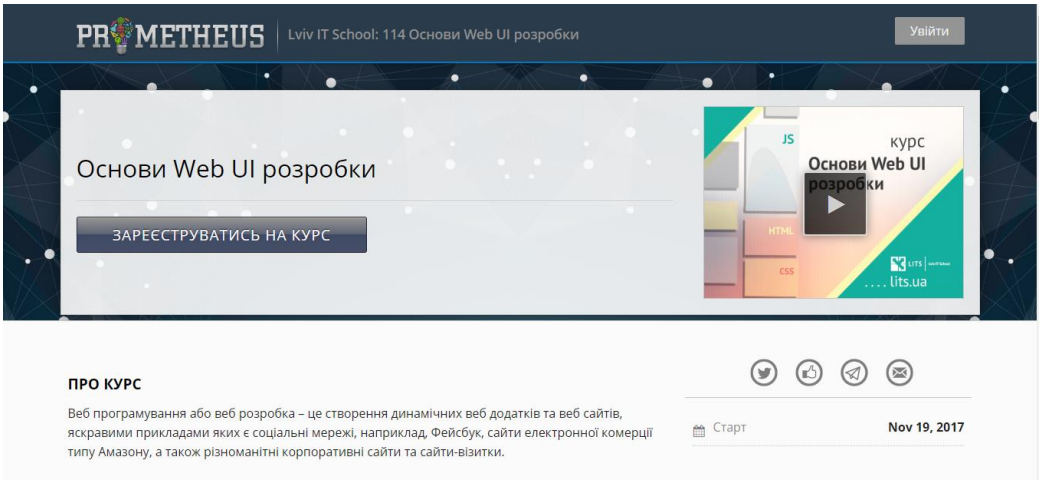

Рисунок 1.6. Сторінка певного курсу веб додатку Prometheus

Провівши невеликий аналіз(табл. 1.2). роботи даної платформи можна зробити висновки. Дана система добре оптимізована. Перевагами є швидкодія роботи серверу, завдяки оптимізованим файлів завантаження JS та CSS ресурсів відбувається в середньому до 1 секунди, як для мобільної версії так і десктопної.

Таблиця 1.2. Результати роботи веб додатку Prometheus

Тип проблеми	Для комп’ютерів	Для мобільних
Час завантаження першого контенту	1,0 сек	4,0 сек
Індекс швидкості завантаження	1,3 сек	4,7 сек.
Час завантаження достатньої частини контенту	1,2 сек	4,6 сек
Час закінчення роботи ЦП	1,2 сек	5,4 сек
Час виконання коду JavaScript	1,1 сек	0,9 сек
Робота системи в основному потоці	0,6 сек	1,1 сек
Розмір структури DOM	141 вузла	141 вузла
Надмірне навантаження на мережу	500 КБ	523 КБ
Помилки в консолі	1	1

Також час завантаження контенту дорівнює 1 секунді, що майже ідеально, це дає змогу користувачеві швидко та без різних перешкод отримувати нові знання в будь якому місті та за допомогою різних пристроїв. Дана платформа добра оптимізована, під всі сучасні браузери, для десктопної версії. До недоліків

можна віднести одну помилку в консолі. Даний веб додаток має не адаптований інтерфейс для мобільних пристроїв, але так як він розрахований на навчання за комп'ютером, то це не є глобальною проблемою.

1.3. Постановка задачі

Ціль в результаті виконання даної магістерської дисертації є розробка та оптимізація швидкості роботи системи управління навчальним закладом. Для отримання необхідних результатів потрібно виконати поставленні задачі:

- Вивчити існуючі рішення та урахувати та використати їх переваги, у
- процесі розробки та оптимізації веб додатку, а також не допустити помилок, що присутні в аналогах;
- Розробити структуру бази даних,
- Визначити основні модулі, та розробити їх логіку;
- Розробити коректну структуру моделей та контролерів, для якісної
- роботи процесу обміну даними;
- Інтегрувати систему кешування даних Memcached;
- Оптимізувати ресурси веб додатку;
- Інтегрувати систему масштабування;
- Провести збір даних швидкодії системи, створення механізму
- автоматичного тестування системи;
- Оптимізація швидкості роботи системи;

Висновки до розділу

В даному розділі було визначено основні питання та цілі роботи. Розглянуто два приклади існуючих веб додатків, а саме Swiftbook та Prometheus. Основними перевагами платформи Prometheus є сучасний дизайн простота користування та ін. Побудована правильна структура та логіка роботи модулів. Основну увагу привертає модуль вивчення курсів який має методичні, відео матеріали та оцінювання кожного завершеного уроку. У результаті проходження курсу кожен

користувач має можливість отримати сертифікат якщо отримує певну кількість балів. Проаналізувавши дані швидкості роботи було зроблено висновок, що головною перевагою даного в додатку є його швидкодія. Завдяки гарним показниками завантаження контенту будь-яка сторінка веб додатку завантажується майже миттєво або доволі швидко при низькій швидкості інтернету, що робить дану платформу доступну майже в будь-якому місті та будь-який час. Також у порівнянні з іншими додатками дана платформа витрачає значно менше часу на завантаження javascript коду та інших ресурсів. Але є деякі недоліки до них можна віднести некоректне відображення мобільної версії в різних браузерах. Що ж стосується іншого веб додатку то про нього можна сказати що він має більше недоліків. Головним недоліком є швидкість загрузки контенту та велика кількість помилок. Визначивши всі необхідні недоліки та переваги існуючих аналогів було поставлено задачі та цілі для виконання магістерської роботи.

2. РОЗРОБКА СИСТЕМИ УПРАВЛІННЯ НАВЧАЛЬНОГО ЗАКЛАДУ

Розробка нового проекту, додатку чи системи завжди починається з проектування бази даних, логіки, клієнтської та серверної частини. Для розробки клієнтської частини системи управління навчальним закладом, вибір припадає на всім добре відому мову розмітки елементів HTML, мову для присвоєння елементів різних стилів – CSS та мову програмування веб-додатків Javascript. Загалом беручи до уваги тенденції використання раніше згаданих мов програмування не є ефективним, тому буде використано нову технологій та спрощені рішення – сучасні фреймворки та бібліотеки, для створення даного веб-додатку а саме:

- Bootstrap 4;
- LeSS;
- ReactJS;

Серверна частина буде створена за допомогою однією з ефективніших концепцій сучасності – MVC, та розроблена за допомогою мови програмування PHP та сучасним фреймоврком розроблений на даній мові – Laravel, який завдяки своїй простоті, універсальності та гнучкості дозволяє створювати проекти будь якої складності починаючи від звичайного сайту та закінчуючи великою, масштабованою системою. Тому саме він добре підходить для вирішення поставленої задачі. . Структура бази

2.1 Концепція побудови структури системи

Концепція MVC (Model-View-Controller: модель-вид-контролер) дуже часто згадується в сучасному світі веб програмування в останні роки. Кожен, хто хоч якось пов'язаний з розробкою веб додатків, так чи інакше стикався з даними акроніма. Сьогодні ми розберемося, що таке - концепція MVC, і чому саме її необхідно використовувати про розробці веб додатку системи управління навчальним закладом . MVC - це не шаблон проекту, це конструкційний шаблон, який описує спосіб побудови структури нашого застосування, сфери відповідальності та взаємодія кожної з частин в даній структурі.

Вперше вона була описана в 1979 році, звичайно ж, для іншого оточення. Тоді не існувало подібної концепції веб додатків.

Шалена популярність даної структури в розробці нових веб додатків склалася завдяки її включенню в два середовища розробки, які стали дуже популярними: Struts і Ruby on Rails. Ці дві середовища розробки намітили шляхи розвитку для сотень робочих середовищ, створених пізніше.

Ідея, яка лежить в основі конструкційного шаблону MVC, дуже проста: потрібно чітко розділяти відповідальність за різне функціонування в веб-додатку, що зображено на рис. 2.1:

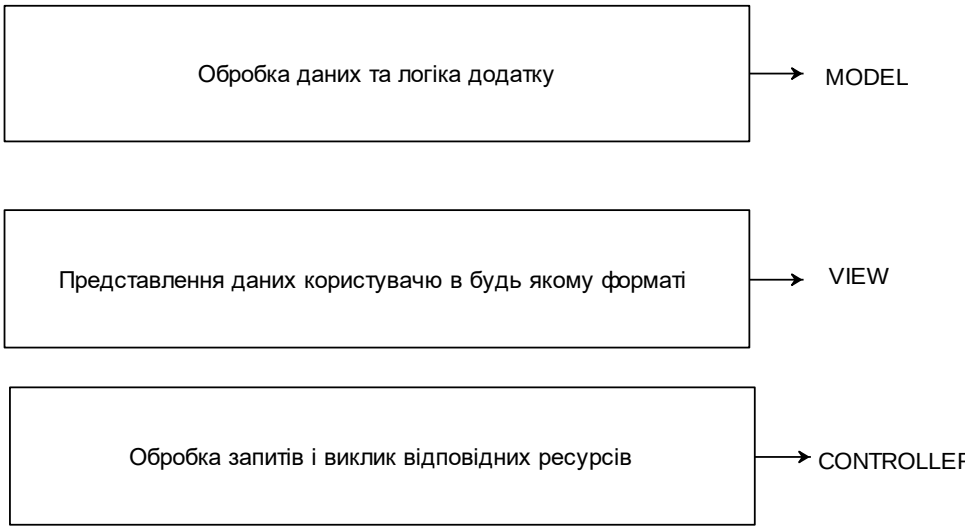

Рис.2.1. Відповідальність кожної частини шаблону MVC

Додаток розділяється на три основних компоненти, кожен з яких відповідає має свою певну роль та відповідає за конкретні поставлені завдання.

2.1.1.Контролер (Controller)

Контролер керує запитами користувача (одержувані у вигляді запитів HTTP GET або POST, коли користувач натискає на елементи інтерфейсу для виконання різних дій). Його основна функція - викликати і координувати дію необхідних ресурсів і об'єктів, потрібних для виконання дій, що задаються користувачем. Зазвичай контролер викликає відповідну модель для задачі і вибирає відповідний вид. У нашому випадку користувач вводить параметри пошуку, які передаються в

контролер, там обробляються та передаються в модель. Звідти ми вже отримуємо дані, що відповідають запиту, які потім будуть виведені в інтерфейсі користувача.

2.1.2.Модель (Model)

Модель - це дані і правила, які використовуються для роботи з даними, які представляють концепцію управління додатком. У будь-якому додатку вся структура моделюється як дані, які обробляються певним чином. Наприклад користувач для додатка представляє собою тільки дані, які повинні бути оброблені відповідно до правил (дата не може вказувати в майбутнє, e-mail повинен бути в певному форматі, ім'я та прізвище не може бути довшим задану кількість символів, і так далі).

Модель дає контролеру уявлення даних, які запросив користувач (інформацію про курс, дані про користувачів, тощо). Модель даних буде однаковою, незалежно від того, як ми хочемо представляти їх користувачеві. Тому ми вибираємо будь-який доступний вид для відображення даних.

Модель містить найбільш важливу частину логіки нашого застосування, логіки, яка вирішує завдання нашого веб-додакту. Контролер містить в основному організаційну логіку для самого додатка.

2.1.3.Вид (View)

Кінцеве представлення системи, тобто те, що бачить користувач. Вид забезпечує різні способи представлення даних, які отримані з моделі, а саме за бази даних. Він може бути шаблоном, який заповнюється даними. Може бути кілька різних видів, і контролер вибирає, який підходить якнайкраще для поточної ситуації.

Веб додаток зазвичай складається з набору контролерів, моделей і видів. Контролер може бути влаштований як основний, який отримує всі запити і викликає інші контролери для виконання дій в залежності від ситуації. Загальний принцип роботи даного шаблону показано на рис. 2.2.

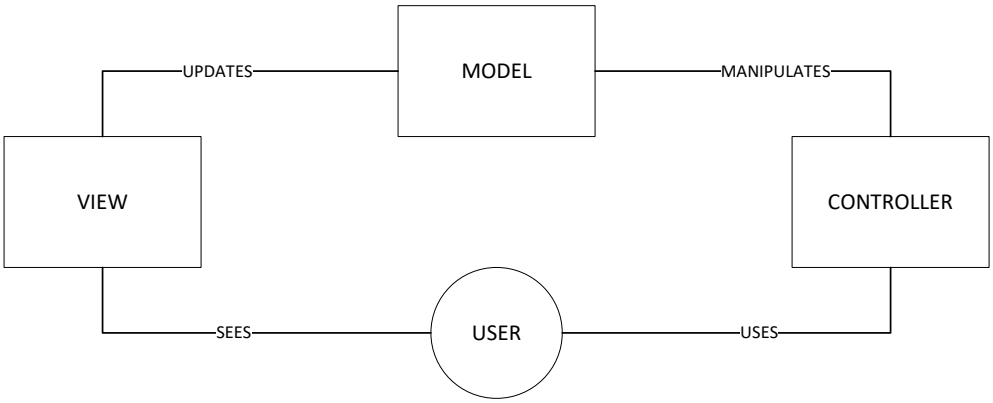

Рис.2.2. Принцип роботи MVC

2.2 Технології та інструменти розробки

В даній частині розглянемо та зроблено вибір технологій, за допомогою яких буде створюватись база даних, клієнтська та серверна частина проекту, порівнявши всі необхідні існуючі інструменти та мови програмування.

2.2.1 Мова розмітки HTML та фреймворк Bootstrap 4

Сучасному світі дуже багато грошей інвестується в розвиток сучасні технології для військової тематики або секретних організацій, тому не дивно, що більшість сучасних технологій які ми зараз на даний момент знаємо були створені секретних лабораторіях саме за таких умов. Не виключенням являється мова позиціювання елементів HTML, яка була створена в стінах секретної ядерної організації у Женеві. А початок великої всесвітньої павутини було закладено відомим британським вченим Тімом Бернерсом-Лі десь наприкінці 1980-х років. HTML можна розшифрувати як HyperText Markup Language або іншими словами мова розмітки гіпертекстових документів. Невдовзі після запуску даної мови одразу ж було випущено перший в світі бруазер тим самим вченим. Це був старт Великого механізму який зараз ми всі знаємо як інтернет.

З часом розвиток технологій не стоїть на місці, тому і HTML мова розвивалася і продовжує це робити до сьогодні. З першої версії мови вже було випущено 8 релізів, які щоразу удосконалювалися задаючи тенденцію розвитку в сучасній веб сфері.

Загалом HTML складається з елементів, які представляють базові компоненти розмітки мови. Як і в будь-якій мові програмування, HTML є свій синтаксис елементів які мають дві основні властивості - це зміст або контент та атрибут. Будь який HTML документ має структуру (рис. 2.3.), що складається з трьох основних частин, а саме:

- декларація типу документу, в якій міститься вся інформація про
- документу;
- Шапка документа, є дуже важливою частиною структури документу
- та містить інформацію, яка не відображається у браузерах і містить технічні відомості. Вся ця інформація знаходиться у парному тезі <head>
- Тіло документу, частина яка відображена користувачеві у браузері,
- містить в собі весь зміст сторінки, який складається з різних елементів.

Кожен елемент представляє собою тег, початок якого відображається як <element-name>, та закінчується </element-name>. Більшість тегів в даній мові є парними, тобто мають тег, який відкривається та закриваються. Але інколи зустрічаються теги які не мають контенту, тому вони не мають кінцевого тегу. По правилам всі атрибути записуються у початковому тезі, а зміст пишеться між початком та кінцевим тегом. Наприклад <element-name element-attribute="attribute-value">зміст елементу</element-name>.

```
<!DOCTYPE HTML>
<html>
<head>
    <meta http-equiv="content-type" content="text/html" />
    <meta name="author" content="TSRh TeaM" />

    <title>Untitled 1</title>
</head>

<body>



</body>
</html>
```

Рис.2.3. Стартова структура HTML документу

Загалом усі елементи відносяться до різні типів розмітки. Елементи структурної розмітки - для опису семантики тексту, якщо сказати іншими словами то ці елементи застосовують для опису текстового контенту. Також існують елементи візуальної розмітки, що призначені для опису різних ефектів

тексту. Одним з найголовніших типів розмітки є елементи розмітки гіпертексту, переходячи з однієї сторінки на іншу, що по своїй суті створюють всесвітню павутину.

Важливу роль відіграють атрибути, що мають відповідну назву та значення які розділені між собою певним синтаксисом та записані у початковому тезі. Завдяки атрибутом мова гіпертекстових документів має можливість зв'язуватися, як з CSS стилями приймаючі документу відповідний сучасний вигляд та іншими мовами програмування, передаю чи отримуючи дані для обробки та зв'язку з сервером або іншими частинами Web структури.

Cascading Style Sheets(CSS) або іншими словами каскадні таблиці стилів. Таблиця стилів були створені з початком розвитку SGML, ще в далеких 1970-х роках. До одної з головних конференції, на якій обговорювались пропозиції щодо стандарту CSS у 1994 році, таблиці стилів не мали певного стандарту. Та в результаті даної конференції було створено групу World Wide Web Consortium W3C, яка вже 1996 році опублікувала CSS level 1 Recommendation, а з часом у квітні 2016 року було опубліковано Cascading Style Sheets Level 2 Revision 2 Робота над яким продовжується і на сьогоднішній день. Завдяки еволюції HTML і CSS веб дизайнери отримали більше можливостей для створення сучасних сайтів та веб-додатків, а користувачів більше якісного контенту. На даний момент за допомогою CSS стилів користувач може побачити легкий, адаптований гіпертекстовий документ на якому з міст або контент будуть чітко зрозумілі для користувачів будь-яких вікових категорій.

Сьогодні завдяки своїм перевагам каскадні таблиці стилів використовуються на всіх сайтах, що дає змогу містити інформацію про всі стилі сайту у одному файлі та при необхідності робити невеликі поправки або редагувати стилі проекту, не змінюючи структуру проекту. Також до переваг можна віднести те, що за допомогою стилів будь-який проект стає адаптивним та може коректно відображатися, як на сучасних так і застарілих комп'ютерах, смартфонах і інших пристроях. Будь-який CSS файл створений з різних стилів,

які потім використовуються в HTML документі присвоюючи до різних елементів. Синтаксис CSS у порівнянні з іншими мовами є доволі простим. Кожен стиль має свою назву яка починається з "." або "#", також є варіант коли не використовується назва стилю, а заміть цього починається з назви тегу до якого даний стиль буде застосований. Далі необхідно мати мінімум одне правило, яке знаходяться поміж фігурних дужок. Ці правила, як і атрибути HTML мови мають назву властивості та її значення, які розділені символом ":" та в кінці крапкою з ";" рис. 2.4.

```
p {
  font-family: Verdana, sans-serif;
}
h2 {
  font-size: 110%;
  color: red;
  background: white;
}
.one {
  color: red;
  background: yellow;
  font-weight: bold;
}
```

Рис.2.4. Структура CSS файлу

Як і час технології не стоять на місці, розвиваючись кожного дня. На допомогу в сучасному CSS приходять різні фреймворки. Це нові бібліотеки створені на основі таблиці каскадних стилів. Тому в даному проекті було прийнято використовувати один з найпопулярніших фреймворків bootstrap4. Даний фреймворк є безкоштовним, універсальним для всіх браузерів. Bootstrap 4 є набором інструментів з відкритим кодом, призначений для створення веб-сайтів та веб додатків. Він містить шаблони типографіки, форм, кнопок, навігації та інших компонентів інтерфейсу, а також додаткові розширення JavaScript. Спрощує розробку динамічних веб-сайтів і веб-додатків. Bootstrap складається з двох частин - наборів таблиць стилів LESS та javascript файлів, які містять різні готові для використання плагіни. Найбільшою перевагою при створенні HTML документу є використання так званої сітки, концепція якої полягає в поділі сторінки на 12 стовбців, вони змінюють свою ширину при різних розмірах екрану , роблячи веб додаток адаптивним будь-які пристрої.

2.2.2 JavaScript та фреймворк ReactJS

Javascript об'єктно-орієнтована мова програмування сучасних веб додатків. З її допомогою легко можна на стороні клієнта працювати з користувачем, збирати, обробляти дані взаємодіяти з браузером, змінювати структуру сторінки. асинхронно відправляти та отримувати дані з серверу. Свій початок javascript бере з 1996 року в результаті стандартизації отримавши назву ECMAScript. Структурно JavaScript складається з трьох основних частин: ядра (ECMAScript), об'єктної моделі браузера BOM та об'єктної моделі документа DOM. Загалом Javascript дещо схожа з мовою C маючи подібний синтаксис рис. 2.5, але також має свої особливості до яких можна віднести функції як об'єкти першого класу, анонімні функції, автоматичне переведення типів та обробка винятків. Одною з головних перевагою даної мови програмування є технологія - використання якої робить сторінки динамічними та зручнішими для користувача.

```
class MyClass {
   constructor () {
   this.myValue1 = 1;
   this.myValue2 = 2;
   }
}

const mc = new MyClass();
mc.myValue1 = mc.myValue2 * 2;
```

Рис.2.5. Синтаксис JavaScript

Javascript дуже легко взаємодіє з документом за допомогою інтерпретатора, що вбудований в браузер. Підключення нового файлу відбувається в шапці документу. На даний момент javascript є одною з найпопулярніших мов програмування, для створення веб додатків та має багато готових відкритих бібліотек для вирішення різних задач. Однією з таких бібліотек є ReactJS - бібліотека основне завдання якої є часткове оновлення контенту веб-сторінки інтерфейсу користувача. React направлений на вирішення великих задач у веб розробці. Бібліотека була створена та представлена доволі недавно у 2013 році на конференції розробників JSConf US, а невдовзі оголошено як програмний продукт з відкритим кодом. React має багато різних особливостей, одна з яких це

одностороння передача даних - процеси відправлення властивостей у render компоненту, як властивості html тега. Компонент може змінювати властивості, які були йому відправлені лише через callback функції, а не напряму редагувати їх. Також однією з переваг є можливість підтримки віртуального DOM, таким чином не розраховувати лише на DOM браузера. Бібліотека знаходить частини DOM, що були змінені, в результаті має можливість відшукати та оновити DOM браузера. React бібліотека має різні методи життєвого циклу, вони інтегруються в проект надаючи можливість обробити дані в різних інтервалах життєвого циклу програмного продукту. Саме тому було прийнято рішення використовувати дану бібліотеку при розробці системи управління навчальним закладом. З допомогою React, який відповідає представленню в концепції MVC, з'являться можливість з легкістю створювати динамічні сторінки, такі як додавання або редагування інформації про студентів та викладачів. Дана бібліотека буде добре взаємодіяти з сервером, отримувати та виводити дані в зручному вигляді для користувача.

2.2.3 PHP та фреймворк Laravel

Серверна частина проекту буде створюватись за допомогою мови програмування PHP. Дана мова програмування є одною з найпоширеніших у веб області, тому має багато готових рішень та бібліотек, для вирішення різноманітних задач. PHP мова яка була розроблена в 1995 році та викладена у мережу Расмусом Лердорфом. З того часу вона еволюціонувала, завдяки чого було випущено нові версії, остання з яких отримав назву PHP 7 у 2015 році. Дана мова може бути інтегрована в будь-який HTML документ та частіше за все за допомогою PHP створюють сервер проекту, або механізми за допомогою яких реалізується звернення до ядра проекту. Одним з найголовніших аспектів чому дана мова є настільки популярною є наявність інтерфейсів та вбудовані бібліотеки для роботи до різних баз даних, що дає змогу з легкістю працювати з інформацією. До таких баз даних можна віднести MySQL, mSQL, Oracle PostgreSQL та інші. Що до синтаксису мови то він дуже схожий на мови C або

Pascal, в результаті чого дана мова є зручною для початку вивчення програмування. PHP дозволяє досить швидко обробляти сценарії, що є дуже ефективним використанням для великих проектів у веб області.

Коли мова йде про безпеку додатку PHP пропонує гнучкі та ефективні засоби для вирішення серйозних питань. Ці засоби діляться на два типи: безпека на системному рівні та безпека додатку. До засобів на системному рівні можна віднести інструменти безпеки, що знаходяться під керуванням адміністраторів, та при короткому налаштуванні дозволяє зберегти безпеку та забезпечити максимальну свободу дій. До засобів безпеки на рівні додатку можна віднести набір функцій за допомогою яких можна реалізовувати різні механізми шифрування інформації, або з легкістю інтегрувати побічні засоби вирішення задачі шифрування.

Однією з головних переваг мови є гнучкість і так як PHP має властивості мови програмування, що може бути вбудована її можна інтегрувати в будь який проект для вирішення різноманітних задач. Також дана мова добре взаємодіє з іншими програмними інструментами та мовами наприклад такі як javascript. Так само як і javascript PHP має свої фреймворки, для вирішення різноманітних задач у веб сфері. До таких інструментів можна віднести CodeIgniter, Symfony, Zend та Laravel. Порівнявши всі переваги та недоліки кожного з трьох років було прийнято рішення вибрати саме Laravel. Він представляє собою безкоштовний фреймворк з відкритим кодом який призначений для роботи з використанням архітектури MVC, що ідеально підходить для вирішення поставлених задач в даному проекті. Свій початок даних фреймовк бере у 2011 році. Маючи певний набір функцій, а саме авторизацію та реєстрація в системі, але з часом Laravel розвивається та набирає популярність. На початку роботи з феєрверком необхідно мати додатковий інструмент який зветься Composer, за допомогою якого можна інтегрувати в будь-який проект різні готові пакети та бібліотеки. Однією з переваг можна віднести безпеку баз даних, завдяки своєму функціоналу даний фреймворк зменшує можливість несанкціонованого доступу до баз даних,

отримати високий рівень безпеки та надійну захист від SQL-ін’єкцій та інших атак різного типу. Завдяки великому функціоналу має можливість створювати проекти будь-якої складності та масштабності з використанням різних інтегрованих модулів та готових рішень.

2.2.4 MySQL

Даному проекті буде використовуватись реляційний тип бази даних та система управління MySQL рис. 2.6. СУБД має багато переваг одне з яких це швидкодія у порівнянні з різними сучасними базами даних, що існують на світовому ринку. Також завдяки своїй простоті вона являється високопродуктивною і відносно простою у використанні та адмініструванні, що полегшує роботу та доступ з різних додатків у будь-який час .

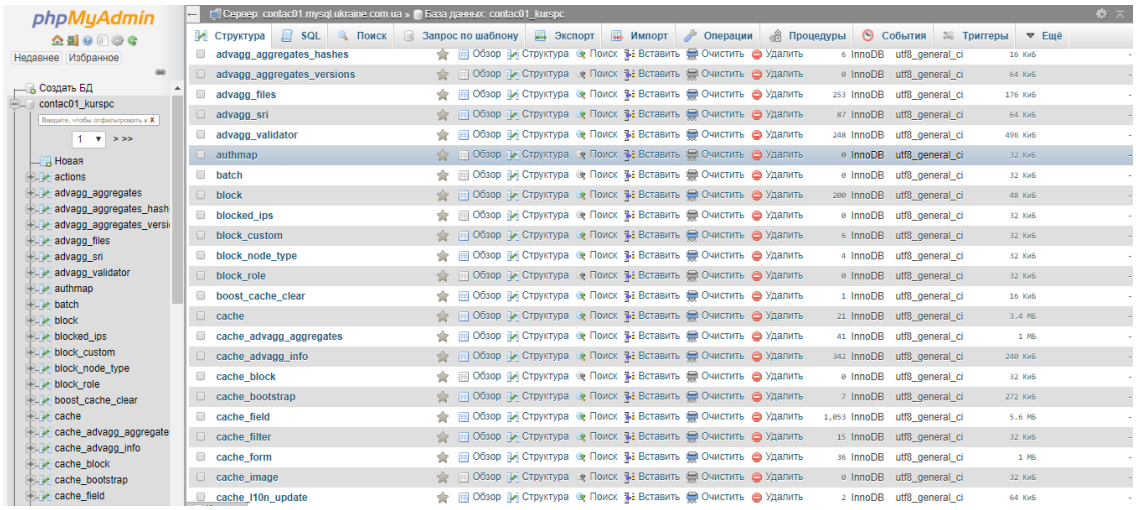

Рис.2.6. Система управління MySQL

2.3 Проектування веб додатку

Проектування даного випадку починається з побудови архітектури. Як і будь-який проект дана система управління навчальним закладом буде мати свою ієрарху підсистем, а саме модулів, які будуть функціонувати окремо та разом давати цілісний, безпечний та сучасний функціонал системи. Кількість модулів рівна 10 і во представлені на рис. 2.7.

Рис.2.7. Структура роботи модулів системи

2.3.1 Проектування модулів веб додатку

Будь який користувач починає роботу з веб додатком саме з модулю авторизації. Для користувача даний модуль відображаються як форма який необхідно ввести дані, а саме електрону адресу та пароль, якщо людина хоче авторизуватись у системі. У випадку невірно введених даних користувача отримує повідомлення про помилку та прохання перевірити дані на коректність. Також у даній формі, а саме у полі "запам'ятати мене" користувач може відмітити чек бокс, та його пароль і особисті дані будуть збережені у браузері, що спростить процес авторизації при наступних спробах. Зазвичай користувачів в систему додає адміністратор, за допомогою іншого модулю управління користувачами, вводячи персональні дані та обираючи тип користувача(студент, викладач, адміністратор), кожен доданий користувач за допомогою адміністратора отримує власний кабінет зі стандартним паролем для входу, який можна змінити при будь-якій нагоді.

Передбачено варіант при якому людина самостійно може зареєструватись в системі, як студент, їй необхідно заповнити форму та отримати дані для авторизації в системі, але навчатись він не зможе поки адміністратор не зробить

його новим учасником групи певного курсу. В даному модулі передбачено функціонал на випадок, якщо користувач не пам’ятає власний пароль. Для того, щоб відновити пароль користувач має перейти на певну сторінку та ввести свою електрону адресу, на який буде відправлення нове посилання на зміну старого паролю та створення нового.

Після додавання нових користувачів у систему наступним модулем є створення групи. Даний модуль є одним з найголовніших модулів веб додатку. Перш за все, для початку роботи даного модуля необхідно створити групу. для цього адміністратор заповнює всі необхідні дані, а саме назву групи, початок заняття та кінець навчання, дату зняття, вибирає філіал в якому буде проводитись заняття, час, аудиторію і найголовніше направлення та курс до, якого буде відноситись дана група. На основі цих даних генерується назва групи, яку адміністратор може бачити в системі. Далі наступним кроком йде вибір основного та резервного вчителя та додавання у групу студентів через форму пошуку, у якій відображені всі студенти системи навчального закладу. У випадку коректного введення всіх даних група буде створена та до неї буде прив’язана персональна папка, для роботи з методичними матеріалами.

В особистому кабінеті адміністратора у модулі створення груп також присутні функціонал фільтрування груп по наступним даним: за направленням, курсу, статусу групи, початку і кінця терміну навчання. В результаті адміністратор отримує список груп де виведено назву групи та кількість студентів в даній групі. Він може спостерігати загальну кількість груп та кількість студентів. що є в даній системі та навчаються. Наступною частиною процесу навчання є модуль методичних матеріалів. Даний модуль має ієрархію направлень та курсів, кожен з яких ділиться на прізвище викладача. В результаті кожен користувач отримує доступ лише для матеріалів власної групи. Результат роботи навчального процесу продемонстровано за допомого діаграми класів рис. 2.8, яка містить об'єкти разом з загальними атрибутами, операціями, зв’язками та семантикою, що є невід’ємною частиною при проектуванні об’єктно орієнтованого веб додатку.

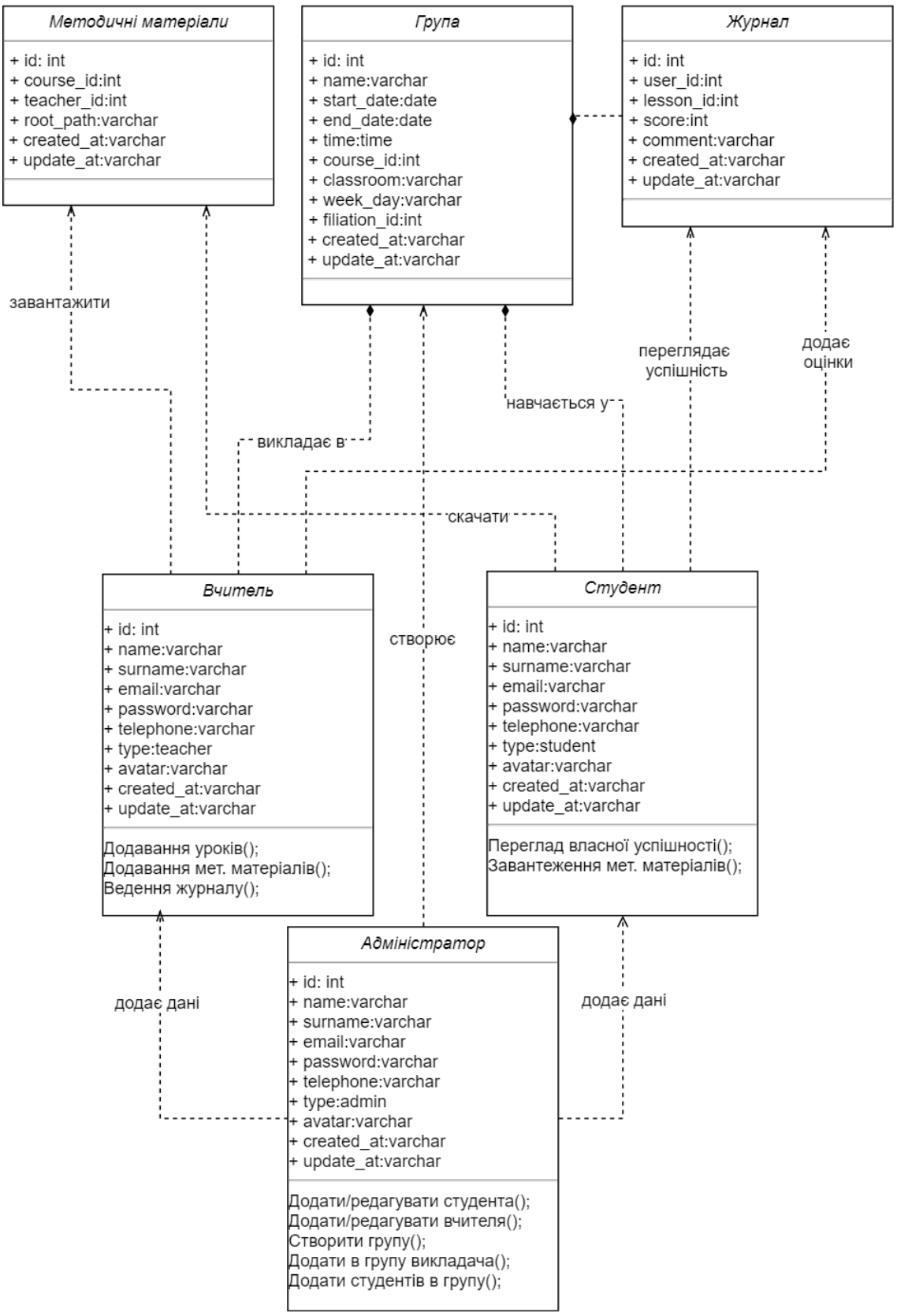

Рис.2.8. Діаграма класів роботи модулю формування груп

Модуль розсилки. Потрапляючи на вкладку даного модулю ми бачимо перед нами поля, для фільтрації електронних адрес студентів та викладачів.

Відсортувавши за напрямом, курсом, статусом та філіалом ми отримуємо список електронних адрес. Створи електронний лист та відправляємо його.

Наступний модуль - це модуль формування документів для друку. Даний модуль, як і попередній складається з двох частин: перша частина це фільтрація за необхідною критеріями, а саме групи та студенти та друга частина - це вибір стовбців, які нам необхідні для створення нового документу. Після проходження всіх етапів користувач натискає кнопку “друк” і отримуємо новий сформований документ з усіма необхідними полями.

Останній модуль, що стосуються навчального процесу - це модуль успішності студенту. Основна функція даного модулю - це слідкування за успішністю навчання кожного студенту. Після закінчення кожного уроку викладач проставляє оцінки отримані за кожним студентом. Студент може отримати оцінку від одного до п'яти в залежності від того, як працював на занятті та як виконав домашнє завдання. Даний модуль тісно зв'язаний з іншим модулем - з модулем обміну подарунків. Бали за весь період навчання студенту додаються та в результаті кожен студент має можливість обміняти загальну кількість балів на певну подарунок, шо може вибрати в модулі обміну подарунків та зв'язатися з адміністратором, для його отримання. Останні модуль додатку - це модуль управління філіалами. Основна концепція роботи модуля управління філіалами, для подальшого використання даної інформації в інших модулях.

2.3.2 Проектування ролей веб додатку

Система управління навчальним закладом поєднує в собі зв'язок людей з системою та іншими людьми. Тому наступним етапом проектування даної системи є користувачі та їх призначення в даній системі - ролі. У даній системі будуть використовуватись наступні учасники, а саме студенти, викладачі та адміністратори. Використовуючи елементи мови моделювання UML було створено діаграму прецедентів рис.2.9 в якій були описані всі можливості для окремої ролі в системі.

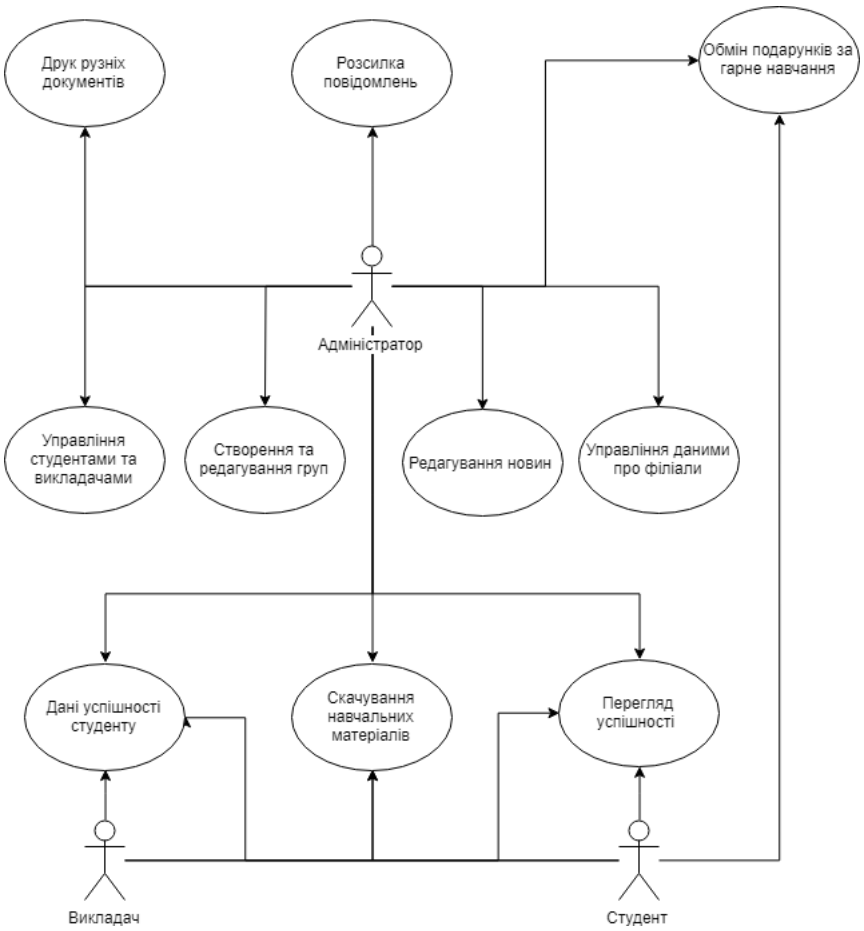

Рис.2.9. Діаграма прецедентів веб додатку

Дана що діаграми прецедентів, за допомогою якої ми маємо можливість представити динамічні або поведінкові аспекти всіх елементів системи, зв'язки між усіма акторами даної системи та функціоналом чи буде створений для коректної роботи. Розглянемо роботу кожного за акторів даної системи. Найбільше кількість зв’язків між процесами та користувачем має адміністратор, саме з ним зв’язані достатня кількість модулів, до них можна віднести: авторизація в системі, управління персональними даними, управління даними про філіал, управління інформацію по студентам викладачам, створення та редагування груп, завантаження та управління методичними матеріалами, додавання та редагування новин, перевірка успішності студентів, підготовка та друк різних документів, розсилка повідомлень, обмін подарунками за успішне навчання, що відображено на діаграмі діяльності адміністратору рис.2.10

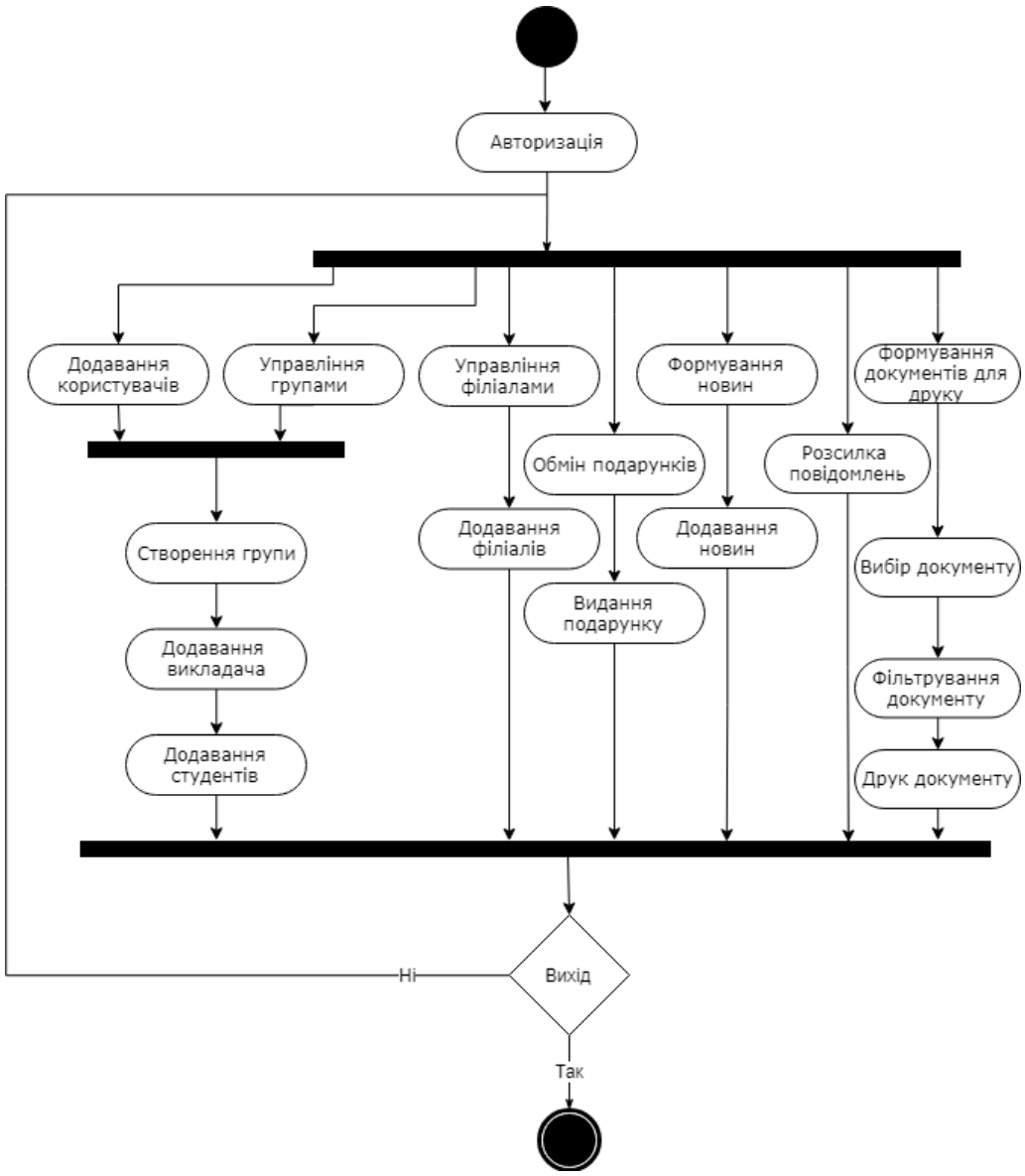

Рис.2.10. Діаграма діяльності адміністратору

Викладач має наступні можливості: управління персональними даними, ведення журналу оцінювання студентів для своєї групи, додавання та управління методичними матеріалами, отримувати листи від адміністратору рис. 2.11.

Студент має такі можливості: управління персональними даними, переглядати інформацію про власні успішність, отримувати листи від адміністратору, скачувати методичні матеріали по курсу на якому він навчається, здійснювати обмін подарунків за успішне навчання рис. 2.12.

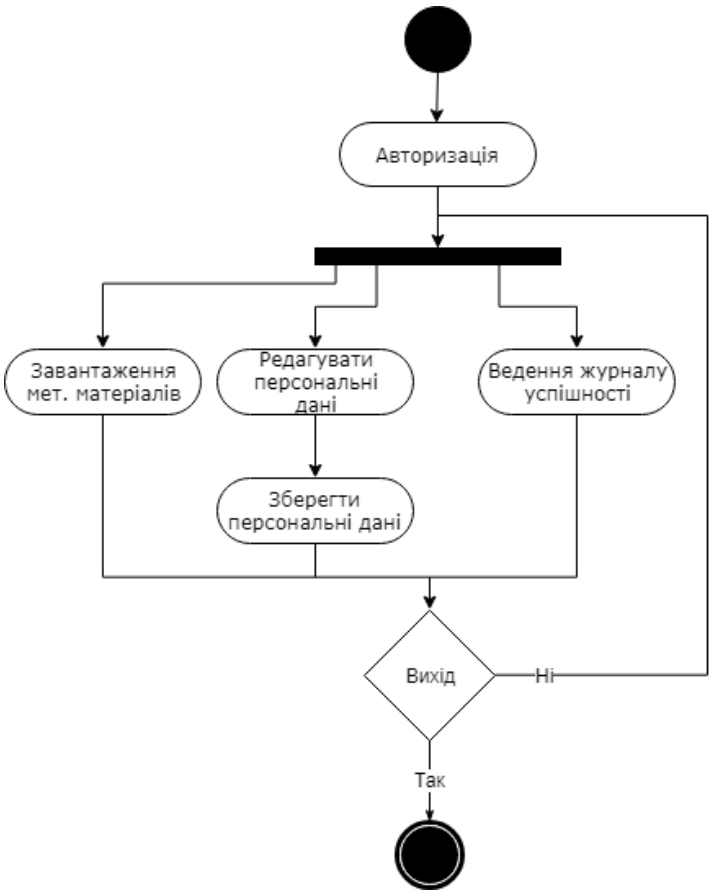

Рис.2.11. Діаграма діяльності викладачу

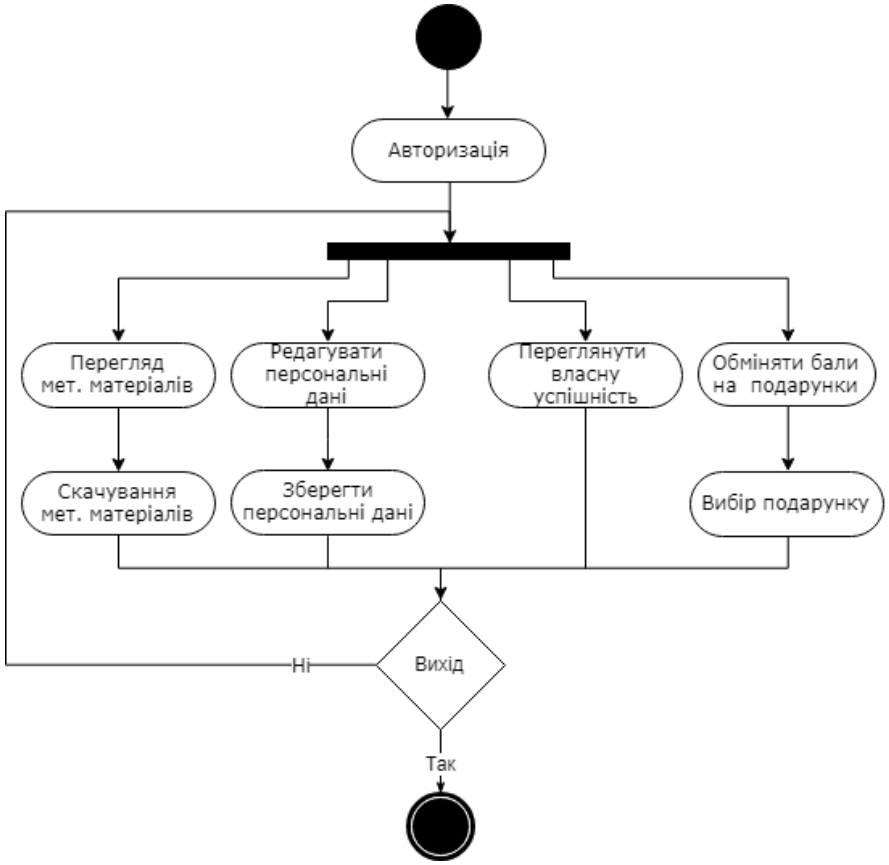

Рис.2.12. Діаграма діяльності студенту

2.4 Структура бази даних

Проектування структури бази даних. На початку процесу створення таблиць, форми та інших об’єктів, необхідно чітко проектувати структуру бази даних. При правильному проектуванні веб додаток буде відповідати всім вимогам та працювати ефективно. Процес проектування бази даних починається з визначення типу бази даних. У даному випадку ми будемо використовувати реляційну базу даних. До наступних етапів можна віднести: формування таблиць, які містить база даних; визначити тип полів у кожній таблиці; формування зв’язків між таблицями та інше.

2.4.1 Опис та зв’язки таблиць

Таблиця 1c_users є одною з найголовніших в структурі бази даних. Вона зберігає мстить дані всіх користувачів системи. Таблиця містить наступні поля: ідентифікатор, прізвище та ім'я користувача, електрона адреса та пароль, до якого буде прив'язано вхід до особистого кабінету, телефон тип користувача рейтинг(для студент) зображення користувача та дані необхідні для оновлення записів.

1c_users_data є таблиця для додаткових даних для користувачів. У таблиці будь приступ присутні наступні поля ідентифікатор, ключ кожного з додаткових полів, його значення, ідентифікатор користувача та поля необхідні для оновлення записів. У таблиці присутні зовнішній ключ для ідентифікатору користувача таблиці 1c_users.

Таблиця в якій міститься загальна інформація про всі курси системи навчального закладу називається 1c_courses. Таблиця містить наступні поля: ідентифікатор курсу, назву і опис, зображення курсу(напрямку), при в'язану папку для початку навчання, статус, батьківські ідентифікатор, для курсів які прив'язані до кожного напрямку навчання, поля необхідні для оновлення записів таблиці. До даної таблиці прив’язаний зовнішній ключ з таблиці груп для ідентифікатору курсу.

Наступною однією з головних таблиць структури бази даних є таблиця 1c_groups. Дана таблиця відповідає за інформацію про групи вона містить наступні поля: ідентифікатор, назва групи, початок навчання, кінець навчання, ідентифікатор курсу, час початку навчання, кімната де проводиться заняття, день, номер філіалу в якому відбуватиметься заняття та дані необхідні, для створення та оновлення записів. Аналогічно і до попередніх таблиць, дана таблиця має поля необхідні для оновлення даних.

Для зв'язку між курсами та користувачами(студентами та викладачем) була створена проміжна таблиця за типом багато до багатьох. Вона містить наступні поля: ідентифікатор, ідентифікатор курсу на який записаний студент, та ідентифікатор користувача, що відвідує певну групу.

Вся інформація про філіали буде міститись у таблиці 1c_filiation. До полів належать: ідентифікатор, зображення філіалу, адреса та телефон, відповідальний за філіал користувач і поля для оновлення записів. До даної таблиці прив'язано зовнішній ключ з таблиці 1c_groups по полю ідентифікатор філіалу.

Таблиця яка також містить зовнішній ключ з таблицею 1c_groups є таблиця 1c_lessons. Дана таблиця створена для ведення успішності навчання студентів. Вона містить такі поля: ідентифікатор заняття, назва та дата проведення. Таблиця є проміжною для таблиць груп та оцінок. Дана таблиця пов'язана с таблицею lc_scores, що містить бали всіх студентів за кожний урок.

lc_root_directories є таблицею, яка необхідна для збереження всіх даних про ієрархію директорій кожного з курсів. В системі управління навчальним закладом є функціонал методичних матеріалів, де розподілена інформація про заняття та домашні завдання по кожному курсу та уроку. Основна ціль даної таблиці зберігати інформацію про зв'язки між директоріями та групами.

Та останньою таблицею в системі є таблиця, що використовується для зберігання даних про подарунків, які користувачі можуть отримати за успішне навчання. А дані про обмін містяться в таблиці 1c_presents_checkout_history. Дана

таблиця має зовнішній ключ для таблиці 1c_presents по ідентифікатору подарунку, що було отримано користувачем.

2.4.2 Створення міграцій

У зв’язку з використанням фреймворку Laravel таблиці будуть створюватись за допомогою міграцій. Міграції – процес схожий на системи контролю версій для бази даних веб додатку. За допомогою міграцій(рис. 2.13) можливий процес зміни структури бази даних, а також є можливість відстежувати зміни, що були зроблені іншим розробником на випадок роботи над проектом в команді. Також за допомогою статичного інтерфейсу до класів Laravel Schema ми маємо можливість підтримки створення та редагування таблиць, не беручи до уваги СУБД яка використовуються.

```php
<?php

use Illuminate\Database\Migrations\Migration;
use Illuminate\Database\Schema\Blueprint;
use Illuminate\Support\Facades\Schema;

class CreateScoresTable extends Migration
{
    public function up()
    {
        Schema::create('lc_scores', function (Blueprint $table) {
            $table->increments('id');
            $table->unsignedInteger('user_id');
            $table->unsignedInteger('lesson_id');
            $table->unsignedInteger('score');
            $table->string('comment');
            $table->timestamps();
        });
    }
    public function down()
    {
        Schema::dropIfExists('lc_scores');
    }
}
```

Рис.2.13. Міграція для створення таблиці оцінок

Міграції представлені, як класи, що описують кожне поле таблиці, його тип, обмеження, зв'язок з іншими таблицями, унікальні та зовнішні ключі рис. 2.9. Для створення нової таблиці або міграції(класів) в Laravel використовується Artisan - команда make:migration. Кожен клас міграцій складається с двох методів up() и down(). Метод up() призначений для додавання нових таблиць, стовбців та індексів, а метод down() призначений для відміни операцій, що виконується попереднім методом. При створенні нової таблиці необхідно використовувати

метод create(). Він приймає два аргументи: ім’я таблиці та завершення, що отримує об’єкт Blueprint, котрий можна використовувати для визначення нової таблиці. Для редагування таблиць та створення нових стовбців необхідно використовувати метод table(). Як і метод create(), метод table() приймає два аргументу: ім’я таблиці та завершення. Додавання нового стовбця містить тип, що вказується при створенні таблиці табл. 2.1.

Таблиця 2.1. Типи даних міграцій

Команда	Опис
$table->increments('id');	інкрементний id (первинний ключ)
$table->boolean('confirmed');	еквівалент boolean для бази даних
$table->char('name', 4);	еквівалент char для бази даних
$table->date('created_at');	еквівалент date для бази даних
$table->datetime('created_at');	еквівалент datetime для бази даних
$table->float('amount', 8, 2);	еквівалент float для бази даних
$table->integer('votes');	еквівалент integer для бази даних
$table->string('email');	еквівалент varchar
$table->time('sunrise');	еквівалент time для бази даних
$table->enum('choices', ['foo', 'bar']);	еквівалент enum для бази даних

База даних системи управління навчальним закладом буде містити 13 таблиць. Повний перелік даних необхідних для створення всіх таблиць наведено у таблиці 2.2.

Таблиця 2.2. Перелік даних для створення таблиць

Назва класу/таблиці	Команда
CreateLessonTable/ 1c_courses	$table->increments('id'); $table->string('name', 255); $table->string('description', 255);

Продовження таблиці 2.2.

CreateLessonTable/ 1c_courses	$table->integer('weight') ; $table->string('promo_folder', 255); $table->string('image', 255); $table->enum('deleted', ['0', '1']); $table->integer('parent_id') ;
CreateUtocTable/1c_users_courses	$table->increments('id'); $table->integer('course_id') ; $table->integer('user_id') ;
CreateGroupTable/ 1c_groups	$table->increments('id'); $table->string('name', 255); $table->date('start_date'); $table->date('end_date'); $table->integer('course_id'); $table->time('time'); $table->string('classroom', 255); $table->string('classroom', 255); $table->string ('short_name', 255); $table->integer ('filiation_id');
CreatePreshistoryTable/ 1c_presents_checkout_history	$table->increments('id'); $table->integer('present_id'); $table->integer ('user_id');
CreateLessonsTable/ 1c_lessons	$table->increments('id'); $table->string('name', 255); $table->date('date'); $table->integer('group_id');

Продовження таблиці 2.2.

CreateFiliationTable/ 1c_filiation	$table->increments('id'); $table->string('filiation_image', 255); $table->string('filiation_name', 255); $table->string('filiation_director', 255); $table->string('filiation_address', 255); $table->string('filiation_ phone');
CreatePresentsTable/ 1c_presents	$table->increments('id'); $table->string('name', 255); $table->date('description'); $table->smallint('price); $table->string('image', 255);
CreateRootTable/ 1c_root_directories	$table->increments('id'); $table->integer('course_id'); $table->integer ('teacher_id'); $table->string('root_path', 255); $table->string('created_at', 255); $table->string('updated_at', 255);
CreateScoresTable/ 1c_scores	$table->increments('id'); $table->integer('user_id'); $table->integer('lesson_id'); $table->integer('score'); $table->string('comment', 255);
CreateUserdataTable/ 1c_users_data	$table->increments('id'); $table->string('key', 100);

Продовження таблиці 2.2.

CreateUserdataTable/ 1c_users_data	$table->string('value', 255); $table->integer('user_id');
CreateTeasersTable/ 1c_teasers	$table->increments('id'); $table->string('name', 255); $table->string('description', 255); $table->string('link', 255); $table->string('image', 255); $table-enum('type',['greet','promo','article']); $table->string('surname', 255); $table->string('position', 255); $table->string('title', 255);
CreateUsergrTable/ 1c_users_groups	$table->increments('id'); $table->integer('user_id'); $table->integer('group_id ');
CreateUserTable/ 1c_users	$table->increments('id'); $table->string('name', 255); $table->string('surname', 255); $table->string('email', 255); $table->string('password', 255); $table->string('telephone', 255); $table->enum('type', ['entrant','student', 'graduate','specialist','teacher_main', 'teacher_reserve','admin','deleted']); $table->integer('rating');

2.4.3 Індекси таблиць

Як і будь-який веб-додаток, що орієнтований на роботу з великою кількістю користувачів, повинен мати активізовано базу даних, для швидкодії. Такий результат можна отримати за допомогою індексів. Ми зможемо зменшити кількість запитів до бази даних та час який буде витрачено на пошук необхідної інформації. Тому необхідно проаналізувати кількість запитів до кожної таблиці, та виділити ти які виконуються найчастіше.

Перший етап взаємодії веб додатку з користувачем це форма авторизації в системі управління навчального закладу. Тому і перша взаємодія з базою даних відбувається на цьому кроці. Виберемо дані, які були введені форму та шукаємо в таблиці 1c_users, що містить інформацію про користувачів, а саме електронну адресу та пароль які донині прив'язаний. Виходячи з цього нам необхідно створити індекс для електронної адреси користувача, але так як він вже є унікальним, він автоматом має індекс.

Наступний функціонал, що потребує великих взаємодій з базою даних є функціонал по управлінню студентами, а саме його пошук необхідного студенту серед всього списку користувачів. Розглянувши перший варіант пошуку бачимо, що в фільтрація студентів відбувається за прізвищами, тобто щоразу система буде перевіряти введений варіант і шукати співпадіння в таблиці користувачів. Таких користувачів у додатку за часом стане багато тому відповідь на запит буде довготривалою, для цього необхідно додати індекс до поля surname у таблиці користувачів 1c_users тим самим, система буде проводити пошук вже не по всій таблиці, а лише в даному стовбці, що з економить час та ресурси бази даних. У другому варіанті пошуку студента за напрямком, курсом та статусу, таблиця до якої звертається система буде невелику кількість даних, тому немає необхідності назначати таблиці індекс так як у майбутньому кількість даних не буде збільшена.

Розглянувши функціонал управління викладачами, можемо сказати, що він аналогічний у порівнянні з функціоналом управління студентам, так як ми

працюємо з однією таблицею та з одними і тими ж полями. І тому як вже було зазначено раніше ми додамо індекс до поля surname у таблиці користувачів.

В системі зараз використовується лише три філіали так, як в майбутньому їхня кількість не буде відносно великою, індекси до даної таблиці використовувати немає необхідності.

Групи – один з найголовніших функціоналів в системі. Головна таблиця з якою взаємодіє даний функціонал, буде мати не багато записів тому фільтрація за напрямком, курсом, статусом та датами початку та завершення навчання групи не є трудомісткою операцією, виходячи з цього можна сказати, що індексів дані таблиці додавати немає необхідності.

Таблиця, що використовується для функціоналу “Успішності” має досить просту структуру. Так як один запис в даній таблиці відповідає оцінці певного студенту на кожному занятті, дана таблиця буде мати багато записів. І тому будь яка взаємодія с нею буде збільшувати час відповіді серверу. При процесі виведення оцінок кожного студенту в особистому кабінету, необхідно перебрати всі записи та вибрати співпадіння, що не є ефективним рішенням. Необхідно додати індекс таблиці 1c_scores до поля user_id. Це допоможе знизити навантаження на базу даних. Що ж стосується інших таблиць в базі даних то можна сказати, що запиту до них не є трудомістким тому як вони будуть мати не так багато даних. Виходячи з цього індекси на даний момент не потрібні у використанні в інших таблицях. Але з часом даний аспект потрібно передивитись так як даних в деяких таблицях може стати більше у результаті чого база даних буде працювати повільніше. Вирішенням даного питання є або додавання нових індексів в залежності від ситуації або архівація даних який не будуть використовуватись.

2.5 Кешування системи

Сьогодні коли користувачів інтернету с кожною хвилиною стає все більше і більше, при розробці веб додатків необхідно приділяти увагу такому аспекту як

кешування даних. Сучасна технологія за допомогою якої відбувається процес тимчасово зберігання даних(файлів, зображень, ресурсів) результатом якого є зменшення часу на відповідь клієнту та зменшення навантаження на сервер. Будь-який великий веб проект повинен мати даний механізм, що допомагає оптимізувати роботу веб додатку.

Веб додатку система управління навчальним закладом використовується технологія кешування – Memcached. З допомогою бібліотеки клієнтської частини PHP, дана технологія дозволяє кешувати дані в оперативної пам’яті одного або більше серверів. Ділення даних відбувається за значенням хеш ключа. Застосовуючи ключ даних, бібліотека клієнтської частини визначає його хеш і віддає його для вибору певного сервера. В даній технології API memcached присутні лише базові функції: обрання серверу, процес установлення зв’язку, видалення, додання та поновлення об'єкта. Для кожного об'єкта є визначають певний час актуальності, розпочинаючи від одної секунди та до нескінченності. При значному використанні пам'яті об'єкти що давно використовувались в кеші автоматично видаляються.

Memcache –це технологія, за допомогою якої відбувається розподілення кешу в пам’яті. Його основний API складається з двох операцій: SET (ключ, значення) та GET (ключ). Memcache подібний хеш-карті, що розповсюджується на декілька серверів, де операції постійно виконуються. Найбільш поширене використання технології Memcache - кешування “важких” запитів до бази даних і рендеринга HTML, щоб ці дорогі операції не повторювалися знову і знову. Щоб використовувати Memcache в Laravel, спочатку необхідно підготувати реальний кеш Memcache. Перш за все необхідно отримати кеш MemCachier від Manifold, який надає швидкий і гнучкий Memcache, сумісний з протоколом memcached. Це створює кеш MemCachier з ім'ям *memcachier* рис 2.14.

```
$ manifold create --product memcachier-cache --plan dev --region us-east-1 memcachier
```

Рис.2.14. Створення кешу

У конфігурації Manifold тепер присутні три змінні конфігурації: *memcachier_servers, memcachier_username* та *memcachier_password*. Також можна знайти їх на аналітичній панелі кеша, до якої є можливість отримати доступ через *sso memcachier*. Щоб налаштувати Memcache в Laravel, ми додаємо наступну залежність в composer.json. Це налаштовує механізм кешування Laravel з MemCachier, що дозволяє використовувати Memcache декількома різними способами:

- прямий доступ до кешу через get, set, delete;
- кешувати результати функцій за допомогою функції rememberforever
- використовуйте Memcache для зберігання сесії
- частинки кеша відображаються
- кешувати всі відповіді

Memcache часто використовується для кешування результатів дорогих запитів до бази даних. Для оптимізації веб додатку необхідно вилучити всі запити з бази даних з повільною операцією. Функція RememberForever дозволяє легко додати кешування в Laravel. Необхідно надати лише два аргументи:

- Ключ кеша
- Функція, яка запитує вашу базу даних і повертає результати

Функція RememberForever шукає ключ в кеші системи. У випадку якщо ключ присутній, повертається відповідне йому значення. В іншому випадку, зазначена функція бази даних викликається. Все, що повертає ця функція, потім зберігається в кеші з відповідним ключем для майбутніх пошуків.

Це означає, що при першому виклику RememberForever викликається дорога функція бази даних, але кожний наступний виклик RememberForever отримує значення з кешу.

Оскільки RememberForever витягує список завдань з кешу, будь-які зміни в базі даних не будуть відображені в списку завдань. З цієї причини щоразу, коли ми змінюємо завдання в базі даних, потрібно зробити кеш недійсним. Кешування

елементів, що обули відображені можливі за допомогою *partalcache*, є можливість кешувати оброблені частини в Laravel. Це схоже на кешування фрагментів в Ruby on Rails або кешування фрагментів в Flask. Якщо у додатку є складні парціальні можливості, рекомендується кешувати їх, тому що відображення HTML може зажадати багато ресурсів процесора. Використання Memcache - добре працює для зберігання інформації для короткочасних сесій. Однак, оскільки Memcache є кешем і тому не є постійним, довгострокові сеанси краще підходять для варіантів постійного зберігання, таких як ваша база даних. У Laravel також легко кешувати всю оброблену HTML-відповідь за допомогою laravel-responsecache. Це схоже на перегляд кешування в Ruby on Rails. Цей пакет простий у використанні і має хорошу документацію в README. Щоб використовувати цей пакет з Memcache, необхідно встановити в config var RESPONSE_CACHE_DRIVER значення memcached.

2.5 Висновки до розділу

Даному розділі було описано технологію що були використані при розробці систем управління навчальним закладом. Розробка клієнтської частини буде проводитись за допомогою фреймворку React, HTML(розмітки елементів), CSS(каскадної таблиці стилів), Bootstrap 4 (CSS-фреймворку), серверна частина буде розроблятись за допомогою сучасного фреймворку, що написаний на мові програмування PHP – Laravel. Також було спроектовано та побудовано архітектуру додатку, яка складалася з різних модулів. Було спроектовано структуру бази даних, до складу якої увійшло 13 таблиць. Кожній таблиці було описані типи даних, для побудови та виконання міграцій фреймворку Laravel, також описані індекси для підвищення швидкості роботи бази даних. Було описано зовнішні зв’язки для кожної із таблиць. Також в веб додаток інтегровано функціонал кешування Memcache, що допоможе зберігати дані на певний термін. Також в системі передбачене масштабування у випадку великого навантаження, в систему будуть додаватись нові сервери для рівномірного розподілу інформації.

3. ОПТИМІЗАЦІЯ ШВИДКОСТІ РОБОТИ ВЕБ ДОДАТКУ

3.1. Оптимізація бази даних

3.1.1 Оптимізація потоків SQL-запитів

У попередньому розділі були розглянуті основні способи оптимізації таблиць та робота СУБД взагалі. У описаних способів є одна спільна риса - загалом вся оптимізація, як правило, проводиться всередині СУБД. Вона може враховувати (і враховує) потік SQL запитів, що йдуть від програми, і, аналізуючи його, може налаштовувати систему таким чином, щоб вона орієнтувалася саме на такий потік запитів і була для нього оптимальною. Але поряд з перевагами такого підходу, можна позначити і дві проблеми. По-перше, у зв'язку з тим, що оптимізація проводиться всередині ядра сервера, ми маємо дуже обмежені можливості впливати на неї, намагаючись розширити з урахуванням деяких припущень. Прикладом таких припущень може бути те, що наша система орієнтована на певний, нестандартний клас задач. Оскільки ми не можемо змінювати саме ядро системи, то ми не зможемо і додати нові, які не використовуються на даний момент способи оптимізації, навіть якщо будемо уявляти собі що саме треба було б для цього зробити. По-друге, хоча ми, як уже було сказано, можемо оптимізувати систему з урахуванням вхідного потоку SQL-запитів, але ми при цьому повинні прийняти його за якийсь еталон, кажучи, що це типовий для програми, яка використовується (додатків) потік. Однак, немає ніякої гарантії того, що сам такий потік оптимальний. Тобто немає гарантії того, що додаток само по собі не генерує такий потік, який далекий від оптимальності. А значить, немає і гарантії того, що, отримавши на вході такий потік, проаналізувавши його, оптимізувавши систему спеціально під нього, виконавши оптимізацію плану виконання кожного запиту з цього потоку в реальному часі і отримавши виграш, скажімо в 20-30%, не існує можливості оптимізувати сам потік і отримати набагато більший виграш. Глобальна оптимізація (оптимізація наборів запитів) неможлива для традиційних реляційних СУБД, інтерфейси яких

припускають почергове виконання запитів. Крім того, цей напрямок зводиться до пошуку загальної частин запитів і обчислення їх тільки один раз. Тобто відбувається поділ складних запитів на більш дрібні частини. У зворотний бік оптимізація наборів запитів не проводиться.

Оптимізація будь-якої системи взагалі повинна починатися з визначення параметрів оптимізації, тобто з визначення того, які характеристики системи для нас найбільш важливі і, відповідно, що ми хочемо отримати в результаті оптимізації.

Коли параметри оптимізації визначено, необхідно визначити найбільш вузькі місця системи, тобто ті частини системи, вплив яких на параметри оптимізації найбільш негативно, і далі оптимізувати в першу чергу їх. Добре, якщо додаток спроектовано правильно і на сервер баз даних видається оптимальний (або близький до нього) потік SQL-запитів.

Однак, немає гарантії того, що вузьким місцем не є додаток, або, вірніше, стик БД і додатку (ми не розглядаємо, скажімо, наявність порожніх циклів при роботі програми додатку, нас цікавлять аспекти, пов'язані з базами даних - час отримання і обробки даних з БД, завантаження сервера і т.п.). Дана робота має на меті виявлення випадків, в яких вузьким місцем системи є стик додатку і БД та опис методики, що дозволяє проводити оптимізацію, орієнтовану саме на такі випадки.

Система, побудована на основі такої методики, повинна буде оптимізувати процеси, що протікають не суто всередині СУБД, а на стику СУБД і додатки. Вона повинна аналізувати вихідну програму і потік SQL - запитів, який вона створює при своїй роботі, структури, які повинні бути видані кінцевому користувачеві (як правило різного виду таблиці), а також саму БД, з структурою і реальними даними, наявними статистиками і індексами і видати рекомендації щодо оптимізації як додатки, так і БД.

3.1.2. Критерії оптимізації

Критерієм оптимізації можуть служити різні параметри. Наша мета полягає в мінімізації часу взаємодії клієнта і сервера. Таким чином, критерієм оптимізації є мінімальність цього параметра (*T*заг).

Розглянемо докладніше з яких складових складається цей час. У найпростішому випадку, коли є всього один запит, процес взаємодії протікає так, як показано на рис. 3.1

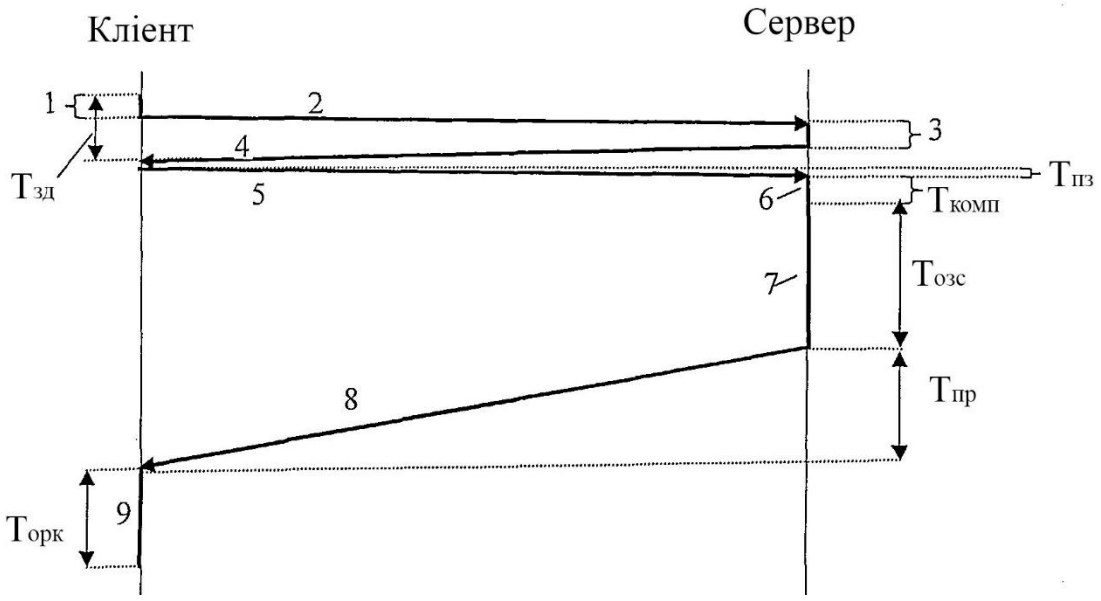

Рис.3.1. Процес взаємодії клієнта з сервером

$$T_{\text{заг}} = T_{\text{зд}} + T_{\text{пз}} + T_{\text{ком}} + T_{\text{озс}} + T_{\text{пр}} + T_{\text{орк}} \quad (3.1)$$

де $T_{\text{заг}}$- загальний час на обробку даних з БД; $T_{\text{зд}}$- це час, необхідний на встановлення з'єднання з сервером БД. Сюди можна віднести створення на клієнті об'єкта, що створює з'єднання, і підтримує його в подальшому (стадія 1), і власне саме встановлення з'єднання, тобто встановлення каналу передачі даних, аутентифікацію (перевірку прав користувача на підключення до сервера), перевірку прав користувача на використання зазначеної їм БД (стадія 2,4) і установку параметрів, таких як формат деяких типів (тип дати, тип грошей і т.п.), реакцію на певні типи помилок і т.п. (Стадія 3).

$T_{\text{пз}}$- (стадія 5) час передачі запиту від клієнта до сервера. Як правило, цей час досить невеликий в порівнянні з іншими накладними запитами. Крім того,

якщо розглядати в якості критерію оптимальності не загальне час обробки запиту, а завантаження сервера, то він взагалі не входить в розрахунок. Однак, просто знехтувати їм не можна. У деяких випадках, можливі ситуації, при яких результат запиту має обсяг, менший самого тексту запиту переданого сервера (наприклад, якщо результуючий набір даних - порожній, або ж повертає одне цілочисельне значення). Якщо створюється потік запитів, що містить безліч таких запитів, а канал передачі даних вузький, то не варто нехтувати цим параметром.

$T_{\text{ком}}$ - (стадія 6) це час компіляції запиту. Передані запити можна розділити на два види. Перший - виклик збережених процедур. В цьому випадку, текст процедур вже попередньо скомпільований і компіляція не потрібно. В цьому випадку час компіляції дорівнює 0. Другий тип запитів - динамічні запити. Текст таких запитів зазвичай збирається під час виконання призначеного для користувача програми. Такий запит під час вступу до сервера повинен бути скомпільованим, тобто побудований план його виконання. Крім того, варто сказати, що при створенні процедури, можна вказувати, що вона повинна пере компілірувати при кожному виклику (це може бути в деяких випадках дуже корисно для оптимізації її роботи). Хоча це і не динамічні запити в чистому вигляді, ми можемо їх віднести також до другого типу, оскільки нас цікавить наскільки великим буде час компіляції. При надходженні динамічних запитів насправді не завжди виконується побудова його плану. План запиту може зберігатися в процедурному кеші. Якщо виявиться що подібний запит деякий час назад виконувався і його план ще зберігається в процедурному кеші, то він може використовуватися для знову надходження запиту. Однак, в будь-якому випадку потрібен якийсь час на пошук в кеші наявного плану. Перед тим як безпосередньо проводити компіляцію, тобто створювати виконуваний план запиту, необхідно провести його лексичний і синтаксичний аналіз. На це для кожного запиту йде якийсь час.

У параметр компіляції входить час, необхідний на лексичний і синтаксичний аналіз, час, необхідний для пошуку плану в процедурному кеші (за

винятком випадків явної вказівки перекомпіляції) і час, що йде безпосередньо на компіляцію (побудова плану) запиту. Якщо рахунок запитів йде на десятки тисяч, то час може виявитися досить великим і проведення оптимізації, шляхом виконання синтезу складного запиту з безлічі простих, може істотно зменшити загальний час взаємодії програми та БД.

$T_{\text{озс}}$ – (стадія 7) час обробки запиту сервером. Це той час, який йде безпосередньо на виконання запиту, причому розуміється, що його план вже побудований. Оптимізацією саме цього параметра традиційно займається оптимізатор запитів.

$T_{\text{пр}}$– (стадія 8) час передачі результату запиту від сервера до клієнта. Якщо канал зв'язку має невисоку пропускну здатність, а обсяг переданих даних досить великий, то цей параметр може стати критичним, тобто складати левову частку часу, що йде на обробку запиту. Це час можна розрахувати як:

$$T_{\text{пр}} = \lambda C \tag{3.2}$$

де λ – пропускна здатність каналу; C – обсяг переданих даних.

Передані дані можуть мати тип фіксованої довжини і тип з динамічною довжиною. Прикладом типу фіксованої довжини може служити int, money, Char(N); прикладом типу даних динамічної довжини може служити Varchar(N), Text, Image. Крім самих даних повинна передаватися службова інформація про заголовках стовпців і їх типах. Службова інформація має різну довжину для різних стовпців, оскільки, імена стовпців можуть бути різної довжини і крім того, для одних з них досить вказати тільки тип, а для інших ще й довжину. Таким чином, можна отримати формулу:

$$C = C_{\text{зфд}} + C_{\text{здд}} + C_{\text{сі}} \tag{3.3}$$

де $C_{\text{зфд}}$ – загальний обсяг даних, що мають типи фіксованої довжини; $C_{\text{здд}}$ – загальний обсяг даних, що мають типи динамічної довжини; $C_{\text{сі}}$ – обсяг службової інформації;

Оскільки дані фіксованої довжини мають одну і ту ж довжину для різних кортежів в одному і тому леї стовпці, але їх довжина може варіюватися від стовпчика до колонку, то можна висловити Cз. ф. д наступним чином:

$$C_{\text{зфд}} = N_{\text{корт}} \sum_{i=1}^{\text{M}_{\text{фд}}} C_i \quad (3.4)$$

де $N_{\text{корт}}$ – число кортежів в результуючому наборі даних; $\text{M}_{\text{фд}}$ – число стовпців фіксованої довжини; C_i – 1-ий стовпець фіксованої довжини (мається на увазі не наскрізна нумерація стовпців фіксованою і динамічної довжини, а нумерація виключно стовпців фіксованої довжини).

Оскільки у стовпців динамічної довжини значення можуть мати різну довжину навіть всередині одного стовпчика, то її можна висловити як:

$$C_{\text{одд}} = \sum_{j=1}^{\text{N}_{\text{корт}}} \sum_{i=1}^{\text{M}_{\text{дд}}} C_{dij} \quad (3.5)$$

де $\text{M}_{\text{дд}}$ – число стовбців динамічної довжини; C_{dij} – фактична довжина 1-ого стовпця динамічної довжини кортежу.

Тоді можна оцінити загальний обсяг переданих даних за допомогою формули:

$$\text{C} = N_{\text{корт}} \sum_{i=1}^{\text{M}_{\text{фд}}} C_i + \sum_{j=1}^{\text{N}_{\text{корт}}} \sum_{i=1}^{\text{M}_{\text{дд}}} C_{dij} \quad (3.6)$$

Можна її висловить і так:

$$\text{C} = \sum_{i=1}^{\text{M}} \left(N_{\text{корт}} C_i * f + (f-1) * \sum_{j=1}^{\text{N}_{\text{корт}}} C_{ji} \right) + C_{ci} \quad (3.7)$$

де M – число стовпців в отриманому наборі даних (M = $C_{\text{дд}}$+ $C_{\text{фд}}$);

$C_i -$ довжина 1-ого стовпця;

$C_i -$ фактична довжина *i*-ого стовпця та *j*-ого кортежу;

$\begin{cases} 1, i \in C_{\text{ф}} \\ 0, i \in C_{\text{д}} \end{cases}$

$C_{\text{ф}} -$ множина стовпців фіксованої довжини;

$C_{\text{д}} -$ множина стовпців динамічної довжини;

Тут нумерація стовпців наскрізна по всьому результуючому набору даних. Варто відзначити, що, маючи статистику БД, можна досить точно оцінити кількість кортежів результату на етапі компіляції1, а значить оцінити параметр

$C_{\text{офд}}$, в той час як оцінити параметр $C_{\text{одд}}$ досить важко. Параметр C_{ci} на момент компіляції відомий зовсім виразно.

Тепер розглянемо, що буде відбуватися при потоці SQL-запитів. Так як клієнтський додаток побудовано розумно тобто на початку програмного модуля створюються об'єкти, необхідні для роботи з БД, встановлюється з'єднання, а потім відбувається вже сама робота, після чого, в самому кінці з'єднання закривається і ці об'єкти знищуються.

При зробленому вище допущенні, отримаємо, що час $T_{\text{зд}}$ буде витрачено тільки один раз - перед виконанням самого першого запиту в потоці, для інших же запитів цей параметр буде дорівнює 0, оскільки з'єднання вже встановлено. Що стосується параметра $T_{\text{орк}}$, то тут питання більш складний. Справа в тому, що обробка різних запитів може йти незалежно, а може і навпаки з обробкою інших запитів. Тому, на даному етапі, просто позначимо сумарну обробку всіх запитів, як $T_{\sum \text{орк}}$.

Що стосується інших параметрів, зазначених у формулі (3.1) для одиночного запиту, то будемо поки вважати, що кожен з них буде повторюватися для кожного запиту в потоці. Тоді, позначивши за L число запитів в потоці, отримаємо наступну формулу для знаходження спільної часу обробки потоку запитів:

$$T_{\text{заг}} = T_{\text{зд}} + L(T_{\text{пз}} + T_{\text{ком}} + T_{\text{озс}} + T_{\text{пр}}) + T_{\sum \text{орк}} \qquad (3.8)$$

3.2. Виділення оптимізуючих класів задач і розробка методу

Вже було сказано про найпростіший приклад, коли одне відношення може бути виведено за допомогою одного запиту, або за допомогою N запитів (на вимогу у кожен рядок), або за допомогою N*M запитів (на кожне скалярне значення). Зрозуміло, що цей приклад явно натягнутий, тобто навряд чи хтось буде писати програму, вибирає кожен рядок окремо. Тепер розглянемо нашу систему управління навчальним закладом.

3.2.1. Запити з зовнішніми ключами

Для опису запитів з зовнішніми ключами(каскадних таблиць) веб додатку введемо кілька попередніх визначень. Для початку введемо операцію зовнішнього з'єднання. Серед стандартних операцій реляційної алгебри немає такої операції, тому доводиться вводити її спеціально.

Результат зовнішнього з'єднання схожий на результат звичайного умовного з'єднання буде, але буде включати всі кортежі з перших відношень, а не тільки ті, для яких у другому з'єднуються щодо знайдені кортежі відповідно до предикату з'єднання. Замість значень атрибутів кортежів других відношень, для яких не виконана умова з'єднання, в результаті будуть потрапляти невизначені значення.

Розглянемо приклад запитів з зовнішніми ключами в нашому веб додатку з функціоналу управління групами де є відношення з таких таблиць як: 1c_filiation , 1c_users, 1c_courses, 1c_groups (фліалу, курсу, групи, студент) результат можна представити у вигляді таблиці (табл. 3.1).

Таблиця 3.1. Каскадна таблиця функціоналу управління групами

Філіал	Курс	Група	Студент
1	15	88	Ковальчук
			Дубина
			Войчик
		100	Анастасенко
	18	141	Курда
			Саленко
		187	Мазнев
			Турчинов
2	22	219	Гарлин
		248	Ронда

Каскадну таблицю можна описати системою відношень.

$$\begin{cases} Col_2 = F(Col_1) \\ Col_3 = F(Col_2) \\ Col_M = F(Col_{M-1}) \end{cases}$$

де M – число стовбців; F –залежність. Це не функція в загальноприйнятому математичному сенсі, це форма запису візуалізації. Вона лише показує, що стовпець, який є аргументом має більший рівень вкладеності (менший порядок угруповання) ніж стовпець, який є результатом залежності. Причому рівень вкладеності аргументу більше рівно на 1. Можна сказати, що якщо для формування таблиці використовується операція $R_1[R_1.A\Theta * R_2B]R_2$ то має місце залежність $Col_1 = F(Col_2)$ ($Col_1 -$ атрибут відношення R_1, $Col_2 -$ атрибут відношення R_2). Якщо ж виконується операція $R_1[R_1.A\ \Theta\ R_2B]R_2$, то має місце або відношення $Col_1 = F(Col_2)$, або залежність $Col_2 = F(Col_1)$. Допускається можливість тієї або іншої залежності з огляду на те, що застосовується операція звичайного умовного з'єднання.

При побудові такої таблиці найбільш простим і природним і найбільш часто вживаним підходом є зробити запит на вибірку з відносини самого верхнього рівня вкладеності (Філіал); потім для кожного обраного кортежу зробити запит на відповідні йому кортежі з відносини другого рівня вкладеності; потім повторити те ж саме для кожного обраного кортежу з цього відносини і т.д.

Для випадків, коли побудова таких таблиць відбувається на локальній машині і обсяг таких таблиць невеликий, це цілком нормальний підхід, що дозволяє найбільш швидко і природно досягти бажаної мети. Однак, коли обсяг таблиці стає істотним(як може бути в нашому випадку), або ж великий обсяг вихідних відносин (у разі коли з вихідних відносин вибираються не всі кортежі), коли запит вибірки кортежів з вихідних відносин є непростим, або ж коли побудова відбувається на веб-сервері, тобто інтенсивність побудови таких таблиць може бути досить великою, то становище змінюється. Справа в тому, що кількість запитів тут буде швидко рости, а значить, будуть зменшуватися час побудови таблиці і збільшуватися навантаження на сервер. Для побудови ж реальної таблиці цілком можуть знадобитися тисячі і десятки тисяч запитів. Як було показано в (3.8), при цьому параметр $T_{заг}$ сильно зросте. Відповідно задачею є зменшення кількості запитів, шляхом оптимізації їх. Отримавши весь набір

даних в одному наборі даних, на клієнті можна обробляти його, врахувавши його структуру. Варто зазначити, що:

а) Процедура обробки дещо ускладниться в порівнянні з описаним вище способом, а значить дещо збільшиться $\sum T_{\text{орк}}$, однак, ці дані можна зменшити за рахунок інших методів оптимізації на стороні клієнту, це збільшення не буде критичним. Параметр $T_{\text{орк}}$, складно оцінювати і зараз ми не будемо цим займатися, залишивши це питання для подальшого дослідження.

б) В отриманому наборі даних матиме місце надмірність. Вона з'явиться через повторень значень стовпців з меншим порядковим номером (з відносини більш високого рівня вкладеності). Поява надмірності веде до збільшення параметра C, а значить і параметра $T_{\text{пр}}$. У деяких випадках воно може і стати істотним.

в) В результаті перетворення SQL-запитів до одного запиту, сильно зменшаються накладні витрати на компіляцію, лексичний і синтаксичний аналіз запитів і їх виконання, тобто узагальнені параметри $T_{\sum \text{комп}}$ та $T_{\sum \text{орк}}$. Що стосується $T_{\sum \text{комп}}$, то при проведенні такого перетворення отримаємо в разі динамічних запитів його зменшення з L* $T_{\text{комп}}$ до $T_{\text{комп}}$. Тобто його зменшення складе (L-1)* $T_{\text{комп}}$. У разі виконання збережених процедур, як уже було сказано вище, зменшення буде менше, оскільки будувати плани не буде потрібно. Що ж до $T_{\sum \text{озс}}$ то питання трохи більш складний.

Для того щоб більш-менш точно оцінити співвідношення сумарного виконання L простих запитів і час виконання одного складного запиту, треба також ввести додаткові параметри. Однак, і без того можна сказати, що час виконання одного складного запиту буде набагато менше.

Таким чином ми, отримали ситуацію, в якій при проведенні запропонованого перетворення одні з доданків формули (3.8) збільшують своє значення, а інші зменшують і потрібно знайти коли таке перетворення проводити доцільно. Схематично основні слідства оптимізації показані на рис. 3.2

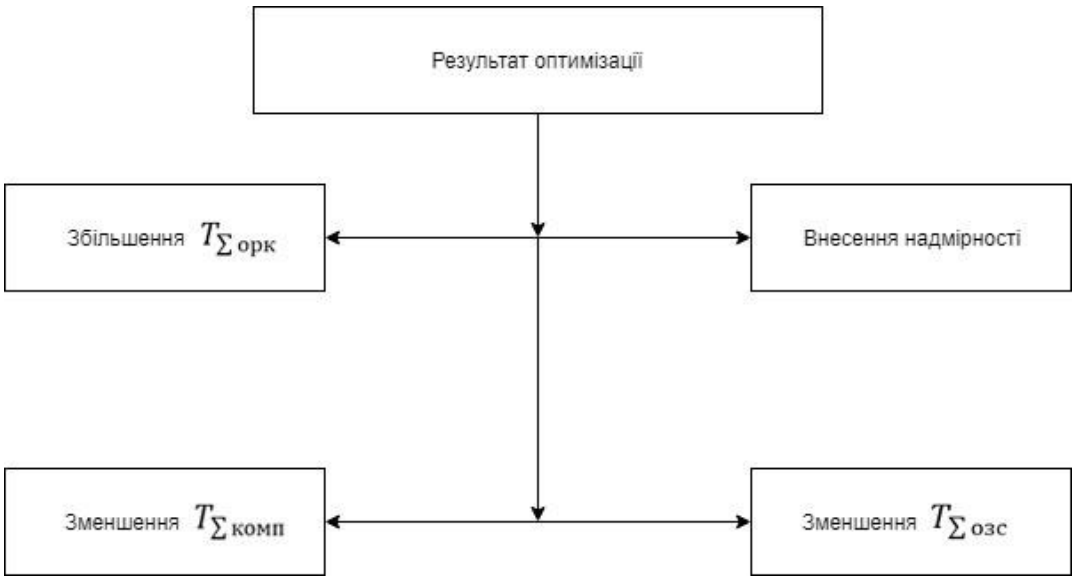

Рис.3.2. Основний результат оптимізації

Використавши раніше описаний метод для оптимізації системи управління навчальним закладом, було проведено аналіз простих запитів, та як результат було створено нові складні запити, результат оптимізації на рис.3.3.

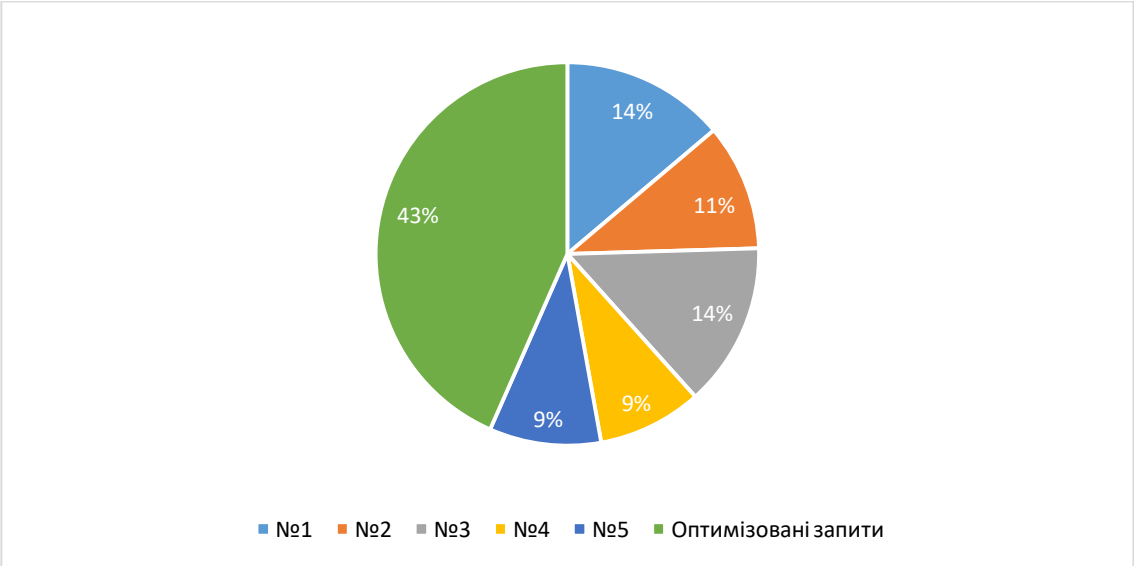

Рис.3.3. Результат оптимізації запитів

Поглянувши на зображення можна зробити висновок, що загальний час виконання запитів було зменшено на 9%.

3.3 Метод кешування індексів

Серед переваг даного алгоритму є те, що він дуже простий в використанні та дозволяє зберігати індекси в ОП. Та він має недоліки - це те, що в пам’ять

попадають індекси, які використовуються надзвичайно рідко та займають там місце, аж поки їх не відтіснять інші. Алгоритм загрузки індексів в оперативну пам’ять системи управління, що базується на методі кешування індексів (МКІ):

1. Для ведення статистики по кількості процесів залучення індексів за період ΔТ СУБД скачує індекси в оперативну пам’ять за звичайним алгоритмом.
2. Після збору статистики за певний інтервал СУБД загружає в ОП лише ті індекси, були використані найчастіше за цей період часу.
3. У випадку, якщо індекс відсутній в ОП, то пошук по даному індексу відбувається прямим зчитуванням його вузлів з дискового індексного простору БД.
4. Після проходження деякого ΔТ в оперативну пам’ять завантажуються лише індекси, шо були використані найчастіше за умови, якщо вони не були скачані раніше.

Такий алгоритм відбувається в два етапи. Перший етап, час якого дорівнює ΔТ , застосовує звичайним алгоритм, адже необхідний час на збір всієї статистики та визначення індексів, що максимально були задіяні за відповідний час. Під час другого етапу в оперативну пам’ять завантажуються лише індекси, що використовувалися найчастіше за попередній ΔТ . Формула для розрахунку:

$$\Delta T = \frac{K_i * T}{K}$$

де Ki – кількість індексів, K – кількість використань індексів, а T – час збору статистики.

Додавши до БД, таблицю, для збору статистки відповідно отримали наступні дані для аналізу. з даними для аналізу використань індексів в нашій базі даних: Ki = 18шт., K = 691 шт., T = 10 годин.

Відповідно: ΔT = 937 ≈ 16хв

Тому, у даному випадку час, за який повинна збиратися статистика буде дорівнювати 16 хв. Відобразимо на рис. 3.4. статистику кількості використаних індексів за кожну годину окремо н протягом нашого інтервалу часу(10 годин).

Пошук, що базується на основі алгоритму, основа якого полягає в методі кешування індексів слід використовувати для баз даних, в яких кількість використань індексів значно вища за кількість самих індексів. Швидкість пошуку при використані даного методу збільшується приблизно на 3%. Також можна підкреслити що навантаження на систем найбільш інтенсивне о 19 та 20 годині, так як кількість звернень до бази даних в цей період найбільша.

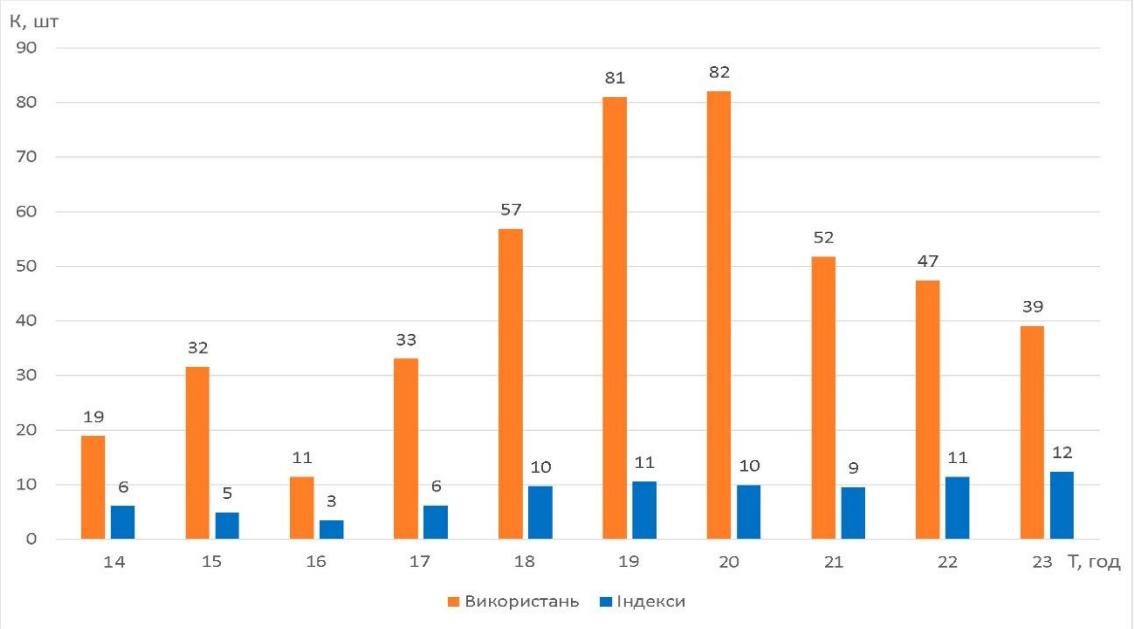

Рис.3.4. Розподіл кількості індексів та їх використань

3.4 Оптимізація клієнтської частини

Оптимізація клієнтської частини полягає в тому щоб зменшити $T_{орк}$, тобто час завантаження та обробки результату сторінки в браузері кожного користувача. Час завантаження залежить від наступних аспектів:

1. Кеш браузера – процес використання кешування зображень, CSS та JS файлів в браузері на тривалий термін, залишаючи розробникові можливість вносити в них зміни без необхідності перейменування файлів, це дасть змогу при повторному завантажені сторінки на скачувати файли, а брати їх з кешу браузера.
2. Об’єднання файлів – об’єднання всіх файли стилів CSS, що

завантажуються на сторінку в один файл, це дасть можливість зменшити час завантаження сторінки браузером за рахунок зменшення необхідної кількості HTTP запитів і підвищення ефективності стиснення, такий самий процес потрібно застосувати для всіх JS файлів.

3. Попереднє розміщення CSS - необхідно розмістити стилі CSS перед файлами JS - це дасть змогу краще розділити їх скачування і збільшити швидкість рендеринга сторінки браузером.

4. Мінімізація файлів - процес зменшення розміру всіх файлів шляхом видалення коментарів і зайвих пробілів, результатом чого є прискорення часу завантаження сторінки браузером.

Провівши аналіз результатів оптимізації клієнтської частини було отримані наступні результати, що покращило швидкість роботи системи. Інструментом для визначання результату було використано консоль в браузері.

Час обробки кінцевого результату головної сторінки менше 1 секунди. Такий результат є задовільним(рис. 3.5)

Name	Status	Protocol	Type	Initiator	Size	Time	Waterfall
personal-office	200	http/1.1	document	Other	3.2 KB	139 ms	
bootstrap.css	200	http/1.1	stylesheet	personal-office	(from disk c...	8 ms	
logo.PNG	200	http/1.1	png	personal-office	(from mem...	0 ms	
font-awesome.min.css	200	h2	stylesheet	personal-office	(from disk c...	14 ms	
start_page.js	200	http/1.1	script	personal-office	(from mem...	0 ms	
fontawesome.min.css	200	http/1.1	stylesheet	personal-office	(from disk c...	13 ms	

21 requests | 8.9 KB transferred | 1.5 MB resources | Finish: 479 ms | DOMContentLoaded: 310 ms | Load: 392 ms

Рис.3.5. Час завантаження головної сторінки веб додатку

Функціонал методичним матеріалів, де користувач має можливість скачати файли для навчання має час завантаження сторінки рівний 1 секунді. (рис. 3.6)

Name	Status	Protocol	Type	Initiator	Size	Time	Waterfall
errors?working_dir=&type=&grou...	200	http/1.1	xhr	jquery.js:9659	982 B	114 ms	
folder.png	200	http/1.1	png	Other	5.8 KB	13 ms	
CWB0XYA8bzo0kSThX0UTuA.woff2	200	http/2+q...	font	Other	(from m...	0 ms	
favicon.ico	200	http/1.1	x-icon	Other	(from dis...	1 ms	
jsonitems?working_dir=&type=&g...	200	http/1.1	xhr	jquery.js:9659	13.6 KB	226 ms	
folder.png	200	http/1.1	png	jquery.js:5868	(from dis...	3 ms	

40 requests | 28.6 KB transferred | 1.8 MB resources | Finish: 1.02 s | DOMContentLoaded: 610 ms | Load: 702 ms

Рис.3.6. Час завантаження сторінки функціоналу методичних матеріалів

Час обробки кінцевого результату сторінки функціоналу управління користувачами також менше 1 секунди(рис. 3.7)

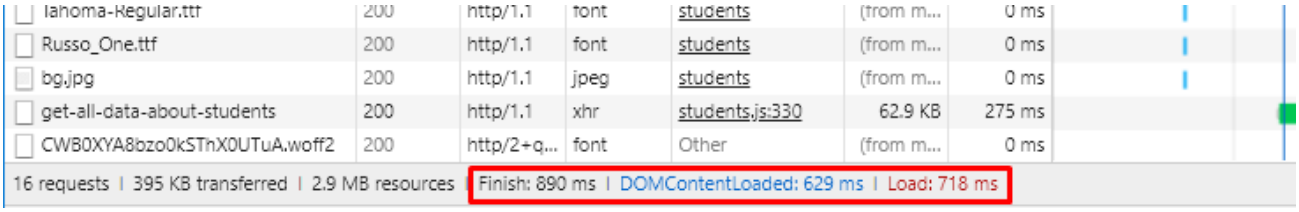

Рис.3.7. Час завантаження сторінки управління користувачами

Аналогічно попереднім даним експерименту час обробки кінцевого результату сторінки функціоналу управління групами менше 1 секунди. Такі результати є задовільними(рис. 3.8)

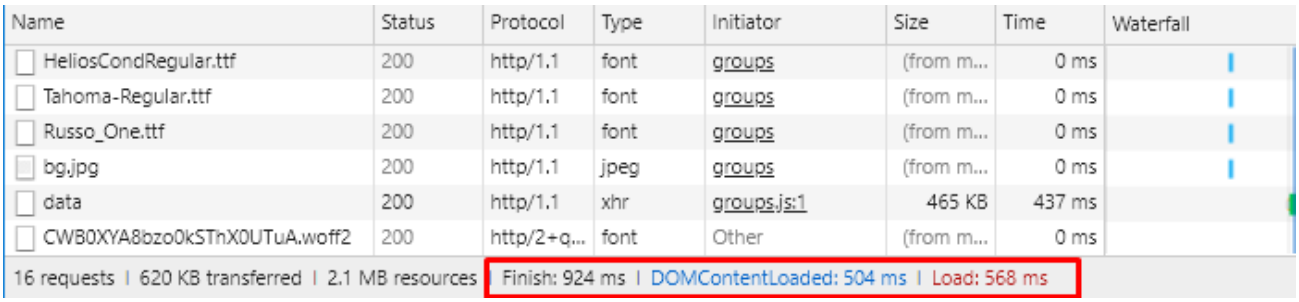

Рис.3.8. Час завантаження сторінки функціоналу управління групами

Провівши аналіз системи управління навчальним закладом на швидкість роботи, було отримані результати, які повністю відповідають всім стандартам компанії Google щодо роботи клієнтської частини. Даний експеримент покращив результати роботи.

Висновки до розділу

Даному розділі було поставлене питання оптимізації та підвищення швидкодії роботи системи. Було докладно розглянуто складові процес з'єднання клієнта з сервером та розбито на різні стадії та визначено час для кожної з них. Як результат оптимізації було необхідно зменшити кожен час з отриманих даних. Перш за все було використано метод оптимізації SQL-запитів. Було проаналізовано SQL запити та визначено з них ті котрі можна було оптимізувати за допомогою складних запитів. Як результат було зменшено час виконання запитів на 9%. Наступним методом було методи кешування індексів.

Проаналізувавши дані ми отримали статистику за допомогою якої було інтегровано даний метод в систему. Також було оптимізовано клієнтську частину за допомогою різних аспектів. Як наслідок було проведено експеримент за допомогою якого було визначено, що час завантаження кожної сторінки було зменшено до однієї секунди що є задовільним за стандартом.

4. МАРКЕТИНГОВИЙ АНАЛІЗ СТАРТАП-ПРОЕКТУ

Одна з основних причин появи, стрімкого розвитку та успішного існування стартапів це важкість та недостатня мобільність потужних підприємств, котрі у своїй діяльності користуються вже усім готовим, та майже не займаються власною розробкою та створенням нових продуктів. Саме тому стартапи, через свою швидкість у генерації та реалізації нових ідей можуть скласти досить непогану конкуренцію великим корпораціям.

Головним джерелом для нового стартапу є вдала свіжа ідея. Саме цікаві та новаторські думки є основним предметом пошуку, на придбання яких зацікавлені сторони не шкодують величезні суми грошей. Навіть сама чиста ідея, не підтверджена жодним матеріальним вираженням, існуючі лише на папері чи на словах (план), коштує вже дуже багато. Ще одним чинником вдалості ідеї служить ступінь її необхідності для споживача, бо сама ідея можу бути цікавою і незвичайною, та якщо вона не нестиме жодної користі, вона нікому не буде потрібна. Стартапи допомагають подолати проблеми, вирішення яких стає можливим завдяки технічному прогресу.

4.1 Опис ідеї проекту

Розглянемо детальніше зміст ідеї, основні напрямки застосування та вигоди користувачів, шо можна отримати при використані додатку(табл. 4.1).

Таблиця 4.1. Опис ідеї стартап–проекту

Зміст ідеї	Напрямки застосування	Вигоди для користувача
Створення веб додатку для управління навчальним процесом та навчання студентів	Використання різними компаніями та навчальними закладами для фіксації успішності навчання студентів, ведення списку, груп, викладачів.	Начальні заклади та компанії мать можливість отримати додаток за допомогою якого можна управляти навчальним закладом та процесом в результаті сучасного та зручного

Продовження таблиці 4.1

		інтерфейсу та багатьох можливостей системи. Користувачі отримують доступ до онлайн навчання, списку методичних матеріалів та журналу з фіксацією їх успішності

Сформулюємо техніко-економічні переваги, шо присутні в даній системі у порівняні з існуючими аналогами. В якості існуючих рішень аналогів до порівняння системи візьмемо веб додаток SwiftBook та Prometheus. Сформуємо преваги та недоліки, а також характеристики нашої системи в порівнянні з визначеними аспектами(табл. 4.2).

Таблиця 4.2. Опис ідеї стартап–проекту

№	Техніко–економічні характеристики ідеї	Продукція конкурентів			W (слабка сторона)	N (нейтральна сторона)	S (сильна сторона)
		Мій проект	SwiftBook	Prometheu			
1	UI дизайн	+	+	+		+	
2	Швидкість роботи	+	-	+			+
3	Створення та управління групами та заняттями	+	-	-			+
4	Забезпечення методичними матеріалами	+	-	+		+	
5	Ведення успішності	+	-	-			+

Продовження таблиці 4.2

6	Кешування додатку	+	-	+		+	
7	Масштабування додатку	+	-	+		+	

Проаналізувавши дані, що наведені у таблиці можна зробити висновок, що даний стартап-проект порівняно із аналогами має переваги, шо робить його конкуренто спроможним на ринку .

4.2 Технологічний аудит ідеї проекту

Проведемо аудит технологій, за допомогою яких можна реалізувати ідею проекту(табл. 4.3)

Таблиця 4.3. Технологічна здійсненність ідеї проекту

№	Ідея проекту	Технології її реалізації	Наявність технологій	Доступність технологій
1	Створення веб додатку для навчання студентів та управління навчальним процесом, а також забезпечення його швидкістю роботи	Мова програмування PHP та фреймворк Laravel	Існує багато рішень для багатьох завдань. Використання фреймворку Laravel забезпечить високу безпеку додатку та простоту реалізації.	У відкритому доступі
		Мова програмування C#	Існує багато фреймворків для реалізації цілі.	У відкритому доступі
		Мова програмування Ruby	Існує багато фреймворків для реалізації цілі.	У відкритому доступі
		Мова програмування Java	Існує багато фреймворків для реалізації цілі.	У відкритому доступі
Обрана технологія реалізації ідеї проекту: мова програмування PHP та фреймворк Laravel				

Провівши аналіз даних з попередньої таблиці, можна зробити висновок, що найкраще для вирішення задачі реалізації підходить мова програмування PHP а саме фреймворк, шо побудований за допомого даної мови Laravel. Завдяки даної технології ми отримаємо можливість реалізації системи управління навчальним закладом. Даний фреймворк має багато готових рішень для вирішення різноманітних задач. Завдяки якісно побудованої архітектури ми можемо побудувати сучасну структуру бази даних що забезпечить високу швидкість роботи додатку. Також великою перевагою є документація даного фреймворку, що забезпечить легку підтримку системи у майбутньому.

4.3 Аналіз ринкових можливостей запуску стартап–проекту

Сформулюємо основні ринкові можливості, що можна використати під час ринкового інтегрування проекту, та ринкові загрози, що можуть перешкодити реалізації стартап-проекту.

Таблиця 4.4. Попередня характеристика потенційного ринку

№	Показники стану ринку	Характеристика
1	Кількість головних гравців, од	>10
2	Загальний обсяг продаж, грн./ум.од	2500 в рік
3	Динаміка ринку	Зростає
4	Наявність обмежень для входу	Немає
5	Специфічні вимоги до стандартизації та сертифікації	Немає
6	Середня норма рентабельності в галузі або по ринку, %	100 %

Провівши аналіз інформації з табл. 4.4. можна зробити висновок, шо даний стартап-проект є привабливим, конкурентоспроможнім та рентабельним проектом виходу на ринок при реалізації поставленої цілі, так як кількість гравців на ринку є невеликою, динаміка ринку щорічно зростає і буде продовжувати це робити, а середня норма рентабельності є максимально великою.

Потенціальні користувачі даного стартап-проекту наведені у табл. 4.5..

Таблиця 4.5. Характеристика потенційних клієнтів стартап–проекту

№	Потреба, що формує ринок	Цільова аудиторія	Відмінності у поведінці цільових груп клієнтів	Вимоги споживачів до товару
1	Реалізації функціоналу онлайн навчання, переведення навчально процесу у цифровий режим, автоматизація процесу навчання	Навчальні заклади будь якого типу університету, школи, дитячого садку, також компанії що проводять перекваліфікацію свого персоналу	Компанії та навчальні заклади зацікавлені у використанні даного веб-додатку, тому що проект дає можливість автоматизації процесу	Основною вимогою є наявність пристрою за допомогою якого є можливість під єднатись до системи через інтернет. Та навики роботи з браузером

Проаналізувавши дані, що наведені у таблиці потенційних клієнтів стартап–проекту можна зробити висновок, що даний проект має багато потенційних клієнтів, але найбільш привабливим ван маж бути для компаній, так як це вже готове рішення, для їх завдань. Якщо мова йде про вищу навчальні заклади даний проект також гарне варіант, але його необхідно допрацювати і інтегрувати більше функціоналу, так як університети це великі структури, які маю багато різних нюансів у начальному процесі.

Визначимо основні фактори загроз, що можуть вплинути на реалізації проекту як продукту на ринку табл. 4.6.

Таблиця 4.6. Фактори загроз

№	Фактор	Зміст загрози	Можлива реакція компанії
1	Зменшення швидкості роботи у зв’язку з великим	В даний момент веб додаток працює у звичайному режимі та витримує навантаження, але якщо кількість	Необхідно звернутись до технічної підтримки, що займається реалізацією даного проекту

Продовження таблиці 4.6

	навантаженням на додаток	користувачів зросте у десятки разів, є ймовірність появи різних проблем,	з проханням виправити проблему, та допрацювати
2	Заповнення бази даних старою інформацією, що може зменшити швидкість роботи	При тривалому часі використання даного веб додатку система буде зберігати дані, що може привести до зменшення швидкодії системи	Розробники компанії мають архівувати дані, додати нові таблиці або просто знищити дані за бажанням клієнту, якщо дані будуть непотрібні.
3	Можливість виникнення питань щодо роботи додатку	У кабінеті адміністратора існує певний алгоритм роботи функціоналу, з яким можуть виникнути питання у користувачів	Компанія має провести ознайомлення з системою та у разі виникнення питань консультувати клієнтів та надавати документацію додатку

Таблиця 4.7. Фактори можливостей

№	Фактор	Зміст можливості	Можлива реакція компанії
1	Удосконалений продукт сучасних рішень	Вихід на ринок, Надання більшого функціоналу що є на даний момент	Вихід нової продукції на ринок; Надання різноманітних типів ліцензій в залежності від потреб користувача; Надання документації; До розробка функціоналу
2	Автоматизація навчального процесу різних закладів	При використані даного додатку оптимізуєтеся та автоматизується процес навчання	Надання консультації у разі звернень різних користувачів
3	Зворотній зв’язок з користувачами	Є можливість отримати дані для покращення продукту	При отримані всіх необхідних даних команді розробників може покращити(інтегрувати) фунціонал

Далі проаналізуємо стартап-проект на ступеневий аналіз конкуренції на ринку табл. 4.8.

Таблиця 4.8. Ступеневий аналіз конкуренції на ринку

№	Особливості конкурентного середовища	В чому проявляється дана характеристика	Вплив на діяльність підприємства (можливі дії компанії, щоб бути конкурентоспроможною)
1	Тип конкуренції: чиста	Існує багато компаній, шо пропонують розробку даного продукту	Перш за все необхідно запустити потужну рекламу. Більш гнучкий підхід до клієнтів, доробка функціоналу.
2	Рівень конкурентної боротьби: національний	Даний стартап проект може бути реалізований по всій території України, в різних компанія та навчальних закладах	За необхідністю інтегрування нового функціоналу при звернені клієнту.
3	Галузева ознака: внутрішньогалузевий	Даний стартап-проект буде використовуватися лише для автоматизації процесу навчання.	Надання чіткої документації по даному додатку, проведення навчання користувачів, та технічна підтримка по роботі з клієнтами. Покращення додатку , випуск нових оновлених, сучасних версій з покращеним функціоналом.
4	Конкуренція за видами товарів: товарно–видова	Дана конкуренція – конкуренція між продуктом одного типу.	Впровадження нового функціоналу, що відсутній у продуктах конкурентів; Надання онлайн підтримки, проведення навчання.
5	Характер конкурентних переваг: цінова	Так як продукт буду однотипний, ціна на нього не буде високою, а якість також буде відповідною	Вести належну підтримку, намагатись у всьому догодити клієнтові. Зробити безкоштовним навчання роботи з даною системою

Продовження таблиці 4.8

6	За інтенсивністю: марочна	Присутність унікального функціоналу, та різних переваг у порівнянні з конкурентами	Впровадження власної назви та власного знаку.

Таблиця 4.9. Аналіз конкуренції в галузі за М. Портером

Складові аналізу	Прямі конкуренти в галузі	Потенційні конкуренти	Постачальники	Клієнти	Товари–замінники
	Основними конкурентами є SwiftBook Prometheus	Різні компанії, що займаються веб розробкою	У даного продукту відсутні постачальники	Навчальні заклади різного типу, та компанії зо проводять навчання свої робітників	У даного продукту відсутні товари-замінники
Висновки	Основною задачею компанії повинно бути надання якісного продукту з високою швидкість роботи, належним функціоналом та інтерфейсом. Надання підтримки користувачів	При збільшені функціоналу додаток має можливість витіснити більшість конкурентів ринку	У даного продукту відсутні постачальники	Деяким клієнтам необхідні зміни в продукту для вирішення персональних задач	У даного продукту відсутні товари-замінники

Проаналізувавши конкуренцію у галузі за М. Портером, Можна зробити висновки що конкуренція у сфері даного продукту присутня, але більшість готових рішень мають свої недоліки. У випадку вирішення та усунення різних недоліків можна отримати доволі велику частку ринку. Одним з головних

аспектів є і підтримка клієнтів, їх навчання, а також постійне вдосконалення продукту. У випадку виявлення всіх недоліків, проект може принести дуже великий прибуток у даній сфері.

Таблиця 4.10. Обґрунтування факторів конкурентоспроможності

№	Фактор конкурентоспроможності	Обґрунтування
1	Збільшена швидкості роботи додатку	Додаток обробляєм та виводить інформацію користувачеві швидше за свої аналоги, тим самим збільшує швидкість роботи користувача з системою
2	Автоматизований навчальний процес	За рахунок присутності функціоналу управління групами заняттями студентами викладачами додаток повністю автоматизує навчальний процес
3	Масштабування додатку	За рахунок використання сучасних технологій масштабування додаток отримає можливість одночасно працювати з великою кількістю користувачів

Таблиця 4.11. Порівняльний аналіз сильних та слабких сторін додатку

№	Фактор конкурентоспроможності	Бали 1–20	Рейтинг товарів–конкурентів у порівнянні з запропонованим						
			–3	–2	–1	0	+1	+2	+3
1	Збільшена швидкості роботи додатку	10		+					
2	Автоматизований навчальний процес	17	+						
3	Масштабування додатку	9			+				

На основі отриманих даних у попередніх таблицях, визначимо SWOT – аналіз(сильні та слабкі сторони, загрози та можливості). Перелік ринкових загроз

та можливостей базується загалом на основі аналізу факторів можливостей маркетингового середовища та факторів загроз.

Таблиця 4.12. SWOT аналіз стартап–проекту

Сильні сторони (S): – Збільшена швидкості роботи додатку – Автоматизований навчальний процес – Масштабування додатку	Слабкі сторони (W): – Складність розробки – Затрати
Можливості (O): – Інтегрування нового функціоналу – Створення додатку під індивідуальні рішення пеаних закладів або компаній – Масштабування додатку	Загрози (T): – Недостаток фінансових вкладень – Поява нових конкурентів

Альтернативи ринкової поведінки або іншими словами перелік заходів, що базуються на SWOT-аналізі. Для виведення та реалізації стартапу на ринок та орієнтовний оптимальний час їх ринкової реалізації з огляду на потенційні проекти конкурентів, що можуть бути виведені на ринок. Сформовані альтернативи аналізуються з точки зору ймовірності отримання ресурсів та строків табл. 4.13.

Таблиця 4.13. Альтернативи ринкового впровадження стартап–проекту

№	Альтернатива (орієнтовний комплекс заходів) ринкової поведінки	Ймовірність отримання ресурсів	Строки реалізації
1	Використання реклами	Використання власних коштів для реалізації рекламної компанії	6 місяців
2	Проведення різних акційних заходів	Головним ресурсом є час та гроші	1-3 місяці
3	Представлення продукту на конференціях інших заходах пов’язаних з даною тематикою	Головним ресурсом є час та гроші	1 рік
4	Надання безкоштовного користування додатком(trial)	Головним ресурсом є час та користувачі	1-3 місяці
5	Укладання угод з великими компаніями для сумісного просування продукту	Висока	2 рік

Провівши аналіз даних у попередній таблиці можна зробити висновок, що найкращим варіантом для ринкового впровадження продукту є надання пробної версії користування стартап-проектом та звичайна реклама.

4.4 Розроблення ринкової стратегії проекту

Таблиця 4.14. Вибір цільових груп потенційних споживачів

№	Опис профілю цільової групи потенційних клієнтів	Готовність споживачів сприйняти продукт	Орієнтовний попит в межах цільової групи (сегменту)	Інтенсивність конкуренції в сегменті	Простота входу у сегмент
1	Навчальні структури пов'язані з інформаційними технологіями. Навчальний процес яких можна реалізувати дистанційно. Студенти, що навчаються в даних закладах.	Даний продукт повністю від цифрує та автоматизує процес навчання студентів	Ліцензія у кількості 1 шт для одної структури навчального закладу	Інтенсивність конкуренції в сегменті висока, так як існує багато різних схожих за функціоналом продуктів	За допомогою багатьох переваг на д аналогами сегмент дає можливість вийти на ринок
2	Компанії. шо проводять навчання працівників та підвищення їх кваліфікації, яким необхідно вести облік працівників. Для великих компаній що мають різні філіали, даний додаток є гарним вибором для вирішення поставленої цілі.	Даний продукт повністю від цифрує та автоматизує процес навчання та підвищення кваліфікації працівників	Ліцензія у кількості 1 шт для одної структури чи компанії	Інтенсивність конкуренції в сегменті висока, так як існує багато різних схожих за функціоналом продуктів	За допомогою багатьох переваг на д аналогами сегмент дає можливість вийти на ринок
Які цільові групи обрано: компанії та навчальні заклади, шо потребують автоматизації процесу навчання свої працівників та студентів					

Відповідно до проведеного вище аналізу можна сформулювати висновок, що для даного стартап проекту є дві цільові групи, яких зацікавить даний продукту та його розповсюдження – начальні заклади та компанії в яких присутній процес підвищення кваліфікації працівників. Що ж стосується стратегії охоплення ринку збуту продукту то було вибрано стратегію масового маркетингу, де діяльність стартап проекту буде орієнтована на навчальні заклади, та компанії методом застосування однорідної композиції .

Таблиця 4.15. Визначення базової стратегії розвитку

Обрана альтернатива розвитку проекту	Стратегія охоплення ринку	Ключові конкурентоспроможні позиції відповідно до обраної альтернативи	Базова стратегія розвитку
Звичайна та технічна підтримка клієнтів, надання функціоналу, що відсутній у конкурентів	Впровадження реклами, надання безкоштовного доступу пробної версії, розповсюдження інформації про функціонал додатку у різні інформаційних ресурси, представлення продукту на різних конференціях, формування великої аудиторії користувачів, залучення партнерів	Певні переваги продукту, збільшення прихильності клієнтів, збільшення прихильності марці	Стратегія диференціації

Далі необхідно визначати базову стратегію конкурентної поведінки стартап проекту табл. 4.16.

Таблиця 4.16. Визначення базової стратегії конкурентної поведінки

Чи є проект «першопрохідцем» на ринку	Чи буде компанія шукати нових споживачів, або забирати існуючих у конкурентів?	Чи буде компанія копіювати основні характеристики товару конкурента, які?	Стратегія конкурентної поведінки
Ні, даний стартап проект не буде першопрохідцем, так як на даний момент існує багато аналогів, з різними недоліками.	Так, основною цілю компанії і збільшення користувачів шляхом переманювання їх у прямих конкурентів	Частково, але головною метою даного стартпа проекту є покращення тих недоліків, що присутні у конкурентів.	Стратегія заняття конкурентної ніші

Таблиця 4.17. Визначення стратегії позиціонування

№	Вимоги до товару цільової аудиторії	Базова стратегія розвитку	Ключові конкурентоспроможні позиції власного стартап–проекту	Вибір асоціацій, які мають сформувати комплексну позицію власного проекту
1	Підтримка з боку розробника	Стратегія диференціації	Постійна підтримка клієнтів що використовують ліцензію. Та підтримка клієнтів для зацікавленості даного продукту	Доступна ціна Орієнтованість на клієнтів
2	Відмінні переваги товару	Стратегія диференціації	Розробка функціоналу, який не має аналогів	Висока якість та надійність

Проаналізувавши дані з попередньої таблиці визначення позиціонування можна зробити висновок, компанія обирає стратегію диференціації, як базову стратегію розвитку.

4.5 Розроблення маркетингової програми стартап–проекту

Таблиця 4.18. Визначення ключових переваг концепції потенційного товару

№	Потреба	Вигода, яку пропонує товар	Ключові переваги перед конкурентами (існуючі або такі, що потрібно створити)
1	Автоматизація процесу навчання студентів різних закладів та підвищення кваліфікації працівників різних компаній	Підвищення якості навчання за рахунок збільшення часу пресу навчання. Не приязність до певних рамок.	Існуючі рішення подібних задач маєте знижену швидкість роботи у порівнянні з даним продуктом. Також відсутність певного функціоналу.

Таблиця 4.19 Опис трьох рівнів моделі товару

Рівні товару	Сутність та складові		
I. Товар за задумом	Автоматизація процесу навчання студентів начального закладу та працівників різних компаній онлайн		
II. Товар у реальному виконанні	Властивості/характеристики	М/Нм	Вр/Тх /Тл/Е/Ор
	UI інтерфейс	-	Висока
	Швидкодія додатку	Інтервал обробки інформації	
	Якість: відповідно загальним стандартам		
	Пакування: ліцензія на веб додаток в кількості 1 шт.		
III. Товар із підкріпленням	До продажу: акційна ціна на обмежену кількість ліцензій		
	Після продажу: цілодобова підтримка клієнтів, удосконалення та ігнорування нового функціоналу		
Стартап буде запатентовано та тим самим захищено від копіювання коду та деяких алгоритмів підвищення швидкості роботи веб додатку			

Після аналізу всієї необхідної інформації наступним етам є визначення меж встановлення цін на стартап-проект

Таблиця 4.20. Визначення меж встановлення ціни

Рівень цін на товари–замінники	Рівень цін на товари–аналоги	Рівень доходів цільової групи споживачів	Верхня та нижня межі встановлення ціни на товар/послугу
–	45-50 тис. грн	від 30 тис. грн	25-60 тис. грн

Таблиця 4.21. Формування системи збуту

Специфіка закупівельної поведінки цільових клієнтів	Функції збуту, які має виконувати постачальник товару	Глибина каналу збуту	Оптимальна система збуту
Залучення партнерів, для кінцевого продажу продукту	Ліцензія	Однорівневий канал збуту	Пряма

Остійнім етапом маркетингового аналізу стартап-проекту е концепція маркетинговій комунікації табл. 4.22.

Таблиця 4.22. Концепція маркетингових комунікацій

№	Специфіка поведінки цільових клієнтів	Канали комунікацій, якими користуються цільові клієнти	Ключові позиції, обрані для позиціонування	Завдання рекламного повідомлення	Концепція рекламного звернення
1	Придбати унікальний функціонал, та отримати повноцінну підтримку від компанії в усіх питаннях	Телефонія, електронна адреса, форми оберненого зв’язку	Підтримка, навчання клієнтів користування системою, якісний унікальний функціонал	Зацікавити клієнта, розповісти про проект та всі переваги які користувач зможе отримати	Класична

Висновки по розділу

В цьому розділі висвітлена основна ідея проекту, проаналізовано потенційні техніко-економічні переваги ідеї, а саме були визначені її сильні, слабкі та нейтральні сторони.

Також був виконаний аудит ідеї проекту і встановлено рівень її технологічної здійсненності. Згідно з отриманими результатами можна зробити висновок, про те що дана ідея – здійсненна. Проведена попередня оцінка ринку стартапу, були визначені потенційні споживачі продукту, та основні загрози та можливості .

Був здійснений ступеневий аналіз ринку, конкуренції на ньому, обґрунтовані чинники конкурентоспроможності, а також проаналізовано сильні і слабкі сторони даного продукту і продуктів конкурентів. Проведений SWOT-аналіз та пошук альтернатив для впровадження продукту. Підсумовуючи отримані результати , можна стверджувати, про те, що даний стартап-проект після впровадження зможе зайняти свою нішу на ринку.

ВИСНОВКИ

У першому розділі було сформовано мету, об'єкт та предмет дослідження, а також область в якій відбувалось дане дослідження. Було проведено аналіз існуючих рішень та розглянуто два приклади сучасних веб додатків. Також було проаналізовано, які переваги має кожна з них такі недоліки. На основі отриманих даних було поставлено завдання розробити нову систему.

А другому розділі було розглянуто які технології були використані для розробки веб додатку, а саме для клієнтської частини та серверної. Було побудовано архітектуру системи, яка складалася з різних модулів, що містили певний функціонал. Також було розроблено структуру бази даних врахувавши всю необхідну інформацію, для підвищеної швидкості роботи з даними. Як наслідок було створено 13 таблиць, в яких використовувалися індекси, для швидкодії, також були описані технології створення міграцій для фреймворку Laravel. Наступним етапом розробки є інтегрування функціоналу кешування та масштабування проекту при збільшенні навантаженні.

На наступному етапу було поставлено питання про оптимізацію веб додатку. Весь процес від запиту клієнта з браузера до його відповіді було розбито на стадії та визначено певний час за який відбувається кожна стадія. Основною задачею було зменшення часу. Першим кроком оптимізації було використання методу оптимізації SQL запитів. Як результат було зменшено час обробки результатів на 9% та було зменшено час компіляції, але в той же час було збільшено час обробки кінцевих результатів. Наступним методом є метод кешування індексів. Було зібрано статистика та проаналізовано і використано метод кешування в індексі в оперативній пам'яті на певний період часу. Оптимізація клієнтської частини була поділена на такі декілька етапів: використання кешу браузера, об'єднання та мінімізація різних типів файлів. В результаті було отримали дані, які задовольняють стандартам.

В результаті виконання даної магістерської дисертації було зроблено оптимізацію та підвищено швидкість роботи системи для навчального закладу.

ПЕРЕЛІК ПОСИЛАНЬ

1. Мова розмітки HTML [Електронний ресурс]: https://uk.wikipedia.org/wiki/HTML
2. Мова стилів CSS [Електронний ресурс]: https://uk.wikipedia.org/wiki/CSS
3. Мова програмування Javascript [Електронний ресурс https://uk.wikipedia.org/wiki/JavaScript
4. Документація мови програмування JavaScript [Електронний ресурс]: https://learn.javascript.ru/
5. Документація Bootstrap [Електронний ресурс] : http://getbootstrap.com/
6. Документація MySQL [Електронний ресурс] : https://www.mysql.com/
7. Дейв Энсор, Йен Стивенсон Oracle. Проектирование баз данных / Лорі, 2006. – 560 с.
8. Маркин А. В. Построение запросов и программирование на SQL / Питер Кому, 2008. – 704 с
9. Харниш, В. Правила прибыльных стартапов : как расти и зарабатывать деньги / В. Харниш ; пер. с англ. В. Хозинского. – Москва : Манн, Иванов и Фербер, 2012. – 279 с.
10. Квашнин А. Как управлять портфелем технологий и интеллектуальной собственностью : серия методических материалов «Практические руководства для центров коммерциализации технологий» / под рук. П. Линдхольма, проект EuropeAid «Наука и коммерциализация технологий», 2006. – 60 с.
11. Ицик Бен-Ган Microsoft SQL Server 2008. Основы T-SQL / БХВПетербург, 2009. – 430 с.
12. Ethan Marcotte Responsive Web Design. Учбовий посібник/ Видавництво Манн, Иванов и Фербер 2012. — 277 с.
13. Algorithms to Large Databases / Bradley, P., Fayyad, U., Reina, C. Scaling , - Proc. 4th Int'l Conf. Knowledge Discovery and Data Mining - Menlo Park, Calif: AAAI Press, 1998.

ДОДАТКИ

ДОДАТОК А

Структура бази даних

ДОДАТОК Б

Архітектура системи для навчання студентів

ДОДАТОК В

Швидкодія системи

ДОДАТОК Г

Результат перевірки на співпадіння